The Dissertation Committee for Vishv Jeet

certifies that this is the approved version of the following dissertation:

Logistics Network Design with Inventory Stocking, Time-based Service and Part Commonality

Committee:

Erhan Kutanoglu, Supervisor

Anantaram Balakrishnan

Jonathan F. Bard

David P. Morton

Leon S. Lasdon

Logistics Network Design with Inventory Stocking, Time-based Service and Part Commonality

by

Vishv Jeet, B. Tech; M. Tech

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2006

To my parents

Acknowledgments

My transformation into a scientist from an engineer has been an interesting process for me. I never thought that I would learn and enjoy so much during my stay at graduate school. My experiences during last four years have been varied and remarkable. The overall development with such a pleasant experience would not have been possible without my advisor's vision and contribution. Erhan's passion, guidance, dedication, and persistence have been constant driving forces for my research interest and motivation. I had the fortune of being Erhan's first Ph.D. student at UT. I took advantage of this fact and had more time with him – several times our meetings went on for multiple hours. During those long meetings we discussed several things including research, personal lives, career, philosophical and societal issues. Erin, Erhan's little daughter, has played a significant part in most of our long meetings every week. Her presence and interruptions used to provide us with much needed breaks in our discussions. Thanks Erin. Erhan's personal warmth and care about my research and overall progress can never be forgotten.

I have been very fortunate to have a chance to work closely with Dave Morton and Leon Lasdon. Their help in my research has been instrumental. When I had just started four years back, Dr. Lasdon told me to try interpolation with GAMS, which was the key to the very first model I developed. The courses of Stochastic Programming and Advanced Mathematical Programming that I took with Dave, changed my thinking fundamentally. Dave was always available in his office without

appointments and all those walk-ins in his office have shaped my research. Dr. Balakrishnan questions during my proposal talk gave me so much insight on my research problem, I am sure it must have taken me at least a year if I were to gather that much insight on my own. I must thank the OR/IE faculty including: Jonathan Bard, Elmira Popova, John Hasenbein, and Wesley Barnes, who taught me several courses and were also available to help whenever needed. A special thanks to John for the help on the last thread of a mathematical proof I was stuck on several months. John upon my request also wrote several columns for our newsletter. People used to read our newsletter because of his column only. Finally, I can never forgot the time I spent with Pedro Balastrassi in his office talking things from all over the world.

I sincerely thank to Diana Ziegler and Ruth Schwab for the wonderful administrative support they have been providing for last several years. At several occasions, they took personal care of my things and paper work I had to do, I can never pay back for the support they provided.

I am grateful to my colleague Amit Partani, whose help in my research has been fundamental. Sometimes his ideas and problem solving skills were awe inspiring. This dissertation would not have taken its current shape without his help. Since my arrival in the United States in fall 2002, I have been extremely fortunate to have a group of people around me whose presence made my life not only easy but also an interesting and memorable one. I am thankful to Ravi Kokku, Arun Venkataramani, Srujana Merugu, Thomas George, Subramanian Iyer, Kedarnath Balakrishnan, Aniket Murarka, Richa Murarka, Dhananjay Adhikari, Amol Nayate, Prem Melville, Joseph Modayil, Ramakrishna Kotla, Kartik Mohanram, Ramadas Nagarjuna, Murali Narasimhan, Neely Mahapatra, Devashree Saha, Haripriya Sridharan, Aditi Partani, Titash Sridharan, Ankur Gupta, Chetan Jhurani and Shilpa Gulati for their time, care, support, and wishes. Among my colleagues at UT OR/IE I sincerely thank Gary Kinney, Thomas Meyer, Dimitri Papageorgiou for their sup-

port and help during first two years when we all were writing qualifying exams and doing course work. I would also like to thank my close friends at UT OR/IE: Clayton Richard, Kranthi Adusumilli, Buke Burak, Güzin Bayraksan, Leonisol Callaos, Xiuli (Sophia) He, and Siwate Rajanasoonthon. Thanks for everything my friends.

Finally, I would like to remember my family: my mother, my brothers and their wives, my sister and her family, and several kids back home who have continued their love and support for all these years. My father, a constant source of inspiration, has always been blessing and helping me from heavens. I have really felt his presence during my difficult times.

Vishv Jeet

The University of Texas at Austin

December 2006

Logistics Network Design with Inventory Stocking, Time-based Service and Part Commonality

Publication No. _____

Vishv Jeet, Ph.D.
The University of Texas at Austin, 2006

Supervisor: Erhan Kutanoglu

Integration and coordination of supply chain decisions is becoming increasingly important in practice and have attracted significant attention from research community. The potential benefit of such integration is clear as more, wider ranges of inherently interdependent supply chain decisions are made simultaneously (i.e., coordinated) for the ultimate optimization of system-wide supply chain performance. However, modeling increasingly more complex problems and developing scalable solution methodologies become a challenge when traditionally separate problems are integrated as the underlying problems themselves are hard. In this dissertation, we model, analyze and develop solution techniques for an integrated network design

problem that simultaneously makes both location/allocation and inventory stocking decisions. The motivation for this problem is post-sales service parts logistics (SPL) in which multiple parts are used to repair multiple products that are in use at geographically dispersed customers. The mathematical model captures important features of real SPL systems: (1) multiple multi-part products with part commonality across products, (2) system-wide, product-level, time-based service requirements, and (3) stochastic demand satisfied by facilities operating with one-for-one replenishment inventory policy. A critical component of the model is the time-based service levels which are functions of both (1) distances between located facilities and customers, and (2) part availabilities (fill rates) of parts, which in turn are functions of stock levels and demand allocations that are being decided as part of the model. In addition to capturing this intricate relationship, our model effectively considers varying fill rates of different parts stocked at various facilities to achieve an overall service level for a product, thus allowing optimal allocation of system-wide product-level service requirements across facilities and parts.

Starting with a nonlinear integer programming model of the integrated problem, we first present a fill rate approximation approach (through piece-wise linearization), which leads to a fully linearized model that can be solved by direct optimization, useful only for small and medium size problems. To facilitate a more effective solution technique for larger problems, we introduce a new variable substitution scheme for the special case of lost sales fill rates, and take advantage of the concavity of a new function in the substituted variables and develop an outer-approximation mechanism. We then develop a specialized relaxation that produces tight lower bounds and a heuristic algorithm that solves a parametric version of the relaxation to obtain provably near-optimal integrated solutions. Based on our analysis with the single part, single product setting, we expand our model and computational study to the cases with multiple products, both with and without

consideration of part commonality. Our extensive computational experiments on variety of problem instances based on industrial data show not only the efficacy of the proposed modeling and algorithmic development, but also the importance of explicit consideration of inventory pooling captured through part commonality. The experiments further show that the improvements through integration of network design and inventory decisions, and through considering part commonality can be significant, which shows the effort behind the overall development is worthwhile.

Contents

List of Tables

List of Figures

Chapter 1

Preamble

This dissertation describes modeling and optimization techniques for an integrated problem of network design with inventory stocking, time-based service level requirements, and part commonality – in low demand (also called slow moving) settings, such as that of service parts logistics systems (Cohen and Lee 1990, Cohen et al. 1997, 1999, 2000a,b).

1.1 Service Parts Logistics

Service parts logistics (SPL) refers to after-market service (providing necessary services, parts, technicians, training, etc.) to address the problems which an existing customer base is experiencing with the products (e.g., electronic items, mainframe computers, servers, or medical equipment, etc.) in use. SPL has become a multi-billion dollar industry and is growing every year. This is partly because high technology product manufacturers are facing with ever decreasing profit margins in sales due to increasing worldwide competition. This is in general true for other industries, but in the high-tech industry, stiff competition leads top manufacturers to

differentiate themselves from competitors by providing an extremely responsive and high-quality after-market service (Cohen and Lee 1990). The responsiveness and quality of services are typically stipulated through service contracts agreed between the manufacturer and the customer. Apart from providing the differentation, these contracts also have great potential to bring in additional revenue and increase customer loyalty through after-market service. Latest surveys show that after-market service accounts for $10\% - 40\%$ of revenue in many industries and up to 50% of inventory investment. In certain industries, 5 to 20 times the initial sale price is spent on subsequent service and consumables (Cohen et al. 1997, 1999, 2000a, 2000b).

1.2 Problem Statement

Given a set of customer locations, their demand for a set of service parts used in multiple products, a set of candidate facilities, part-product relationships, and part commonalities, we seek to design and stock a network of operating facilities and customers to deliver required time-based service levels for each product with the minimum total system cost.

1.2.1 Time-based Service Levels

Time-based service levels are a representation of long-term (or steady-state) service rendered to the customer – they are used to evaluate compliance with the service agreements. Although, in general, it is difficult to estimate the true value of the services rendered in a specified period, yet an accurate assessment is essential for system-wide planning and optimization. The customer service agreements spell out specific service levels (in terms of regarding response time, part availability, and overall quality of the service) to be achieved. For example, a typical service level

requirement may say: 70% of demand for service parts for a specified product must be satisfied within 4 hours from the time the part request is made[1].

1.2.2 Challenges

The main source of challenges in designing, stocking, and operating an SPL network is to guarantee an extremely responsive service so that customers' mission-critical systems are up and running again.

- *Time-based services:* SPL systems are set up to deliver certain service levels jointly agreed contracts between the customer and the manufacturer. The main challenge, however, is to provide the service within a time-window – measured in hours – which necessitates employing a network with facilities closer to critical customers.

- *Integrated decisions:* The requirement to achieve above mentioned service levels puts the problems of network design and inventory stocking, hitherto considered two separate problems, into a single problem. Note that the service levels are defined for products but the inventory is stocked for service parts (that make up the products). We can achieve a target system-wide product-level service through different parts stocked at multiple facilities. Therefore, the associated optimization problem is a two-dimensional service allocation problem, i.e., allocation of a time-based service level (1) across facilities and (2) across parts. Since locations of the facilities and their fill rates together determine what service level a network can deliver, there is an interdependency between network design decisions and inventory stocking decisions. Due to

[1]This would also require an appropriate planning for tools, technicians, and vehicles but in this dissertation we address the problem for service parts stocking only.

the traditional hierarchical segmentation of strategic and tactical decisions, one could be tempted to design the network first (at the strategic level) and then to decide the stock levels for the designed network. However, the decoupled sets of network and inventory decisions may lead to high costs as the inherent interdependency between the decisions would be lost. Furthermore, the network design decisions are increasingly becoming a tactical issue, pushing them closer to the inventory decisions in the hierarchical framework. This is mainly due to redesign flexibility that comes with the warehouse and transportation activities outsourced to third-party logistics services providers. Hence, making and revising simultaneous network design and inventory decisions more frequently is becoming a reality, which opens the way for integrated decision making and overall optimization.

- *Part Commonality:* In general, a part goes into and becomes a component in many products, due to modularity of product designs and standardization of parts part commonality is prevalent. In SPL modeling and optimization, one has to decide how to handle the part commonality. An easy way is to ignore part commonality, assuming that each part-product combination is unique, as if a specific part-stock level is "reserved" for each product. Another, rather more realistic and complete modeling effort would be to explicitly consider part commonality, and have a stock level for each part to be used to serve all the products in which the part is common. Considering part commonality within the integrated decision framework will not only make the model more realistic, but also lead to further reductions in the total system costs because of its risk pooling effect. However, modeling part commonality and its effect on system behavior explicitly along with integrated decisions of network design

and inventory stocking is another challenge.

- *Common Network:* In general, an SPL system has a single network of stocking facilities that serves geographically dispersed customers. Each customer typically has multiple products in use. Each product, in use at multiple customer locations, consists of various parts, and has its own dynamics of demand for service parts. Each facility in the network stocks these service parts to repair multiple products that are in use across many customers. Each product has a system-wide service level to be achieved when all the customer demands for all the parts that go into the product. Designing and stocking a network, which is shared across multiple products and customers, to satisfy product-level target service requirements is a significantly difficult problem, especially when compared to a simpler version, where a single part, single product network is considered, since the common network must be feasible for all products under consideration.

The above list of challenges is not complete but only the list of challenges addressed in this dissertation, we provide more discussion on challenges in SPL environment in future research section at the end.

1.2.3 Contributions

To address the challenges highlighted in Section 1.2.2, we introduce the following that also summarize our main contributions:

- *Modeling:* We model the overall integrated problem with mild assumptions. To the best of our knowledge integrated network design and inventory stocking models with variable fill rates and with high-degree of service allocation

(achieving a system-wide service level across multiple parts, facilities, and customers) do not exist in the literature. One exception is Candas and Kutanoglu (2006a) who show the benefits of considering inventory decisions in network design problems. They linearize the nonlinear fill rates using step functions to solve moderate size problems to show the benefits of integration. There have been other multiple attempts to combine simple facility location models with inventory stocking models, but time-based product-level service levels, and part commonality have not been considered explicitly. The service allocation problem is explicitly considered in our models, i.e., instead of assuming a fixed and constant fill rate for all parts and facilities, we let the model decide the fill rates, demand allocations, and stock levels.

- *Solution Methodology:* The integrated problem for a single service part used in just one product becomes a complex mixed integer nonlinear program (MINLP), which presents formidable challenges in solution methodology. We propose a variable substitution scheme which not only convexify the problem but also allows us to use an effective outer-approximation of nonlinear functions. We also propose a relaxation scheme that leads to tight lower bounding. The relaxation scheme further leads us to develop an upper bounding scheme that successively solves a series of the parametric version of the relaxed problem. The integrated problem with part commonality across products has an additional complexity for exploiting the part commonality. We handle this issue in our modeling as well as in solution scheme effectively.

- *Computation:* We conduct extensive computational experiments on variety of problem instances (both randomly generated and based on real data) and show that the overall approach is quite effective. We also compare our computa-

tional results with the conventional approaches such as the two-stage solution technique and also with the approaches that ignore part commonalities.

- *Insight:* Having solved the problem effectively, we emphasize the insights obtained from computational results, which are further useful to solve the problem in even larger perspective. Specifically, we identify an underlying fractal structure in the problem which can be effectively exploited to model "inventory sharing" across facilities and customer centric service levels. We identify several other relaxation schemes which are useful in cases where the nonlinear functions in the formulation may not have nice properties like concavity. The overall analysis gives a complete picture of issues related to integrating decisions in network design and inventory stocking.

1.3 Related Work and Literature

Network design and facility location models are among the most studied models in the operations research literature. Magnanti and Wong (1984) review the early literature on the facility location problems, Drezner (1995) gives a survey of applications and methods for facility location. Daskin (1995) is a reference text on discrete facility location problems. The research relevant to the work in this dissertation are the studies based on the uncapacitated facility location (UFL) model that consider time-based demand/service coverage restrictions. Goldberg and Paz (1991) and Nozick (2001) are two close examples. More recently, Snyder and Daskin (2005) investigate the UFL problem with potentially unreliable facilities and Daskin et al. (2005) discuss facility location problems in designing a supply chain.

Given the vast literature in inventory related research, we limit our focus to the studies related to multi-location inventory management models that explic-

itly consider service level constraints for low-demand service parts. Early work in this area includes Sherbrooke (1968, 1986), Muckstadt (1973) and Muckstadt and Thomas (1980). More relevant studies include Verrijdt and de Kok (1995, 1996) who discuss distribution planning with service level constraints. Ouyang and Wu (1997) propose an inventory model with variable lead times and service level constraints, and Song (1998a) investigates a simplified time-based service level with base stock policies. Cohen et al. (1988) study inventory systems with service constraints and priority demand classes. Studies related to successful implementations in service parts logistics systems include Cohen et al. (2000a) in automotive, Cohen et al. (1988, 1990, 1999) in computers and other electronics, and Rustenburg et al. (2001) in military logistics. Sherbrooke (1992) provides an overall review of multi-echelon inventory management from a military perspective with a focus on repairable parts. More recently, Muckstadt (2005) has become a new reference on general SPL models and algorithms. Zipkin (2000) is a recent reference on inventory theory.

Integration of network design and inventory stocking has been under investigation for more than 25 years, with an increased attention more recently. The earliest references include Geoffrion (1979), Geoffrion and Roy (1979) and Geoffrion and Powers (1980). Geoffrion (1989) specifically discusses both modeling and solution techniques for integration. Ignoring transportation, Wagner (1974) and Cohen and Lee (1988) study integration of production and inventory systems for multi-facility and multi-warehouse companies. Barahona and Jensen (1998) are perhaps first to study a problem that combines discrete fixed-charge facility location and inventory more explicitly. Their model has simple inventory decisions (to stock or not to stock certain parts) along with location decisions and their solution technique uses Dantzig-Wolfe decomposition. Nozick and Turnquist (1998) study impact of

integrating inventory into a fixed-charge location model assuming fixed fill rates. Nozick and Turnquist (2001) propose an approximate inventory cost function embedded in the fixed cost term of a facility location model, which is an extension of the fixed-charge UFL problem. They model two versions of the problem, one minimizing the total logistics costs and the other maximizing the service coverage. In another related study, Nozick (2001) considers service coverage constraints with limited consideration on inventory. More recently, Daskin et al. (2002) and Shen et al. (2003) study a single-part joint location-inventory model. Their models consider an economic order quantity (EOQ)-based ordering policy and constant fill rates across all facilities, which make the models more suitable for high demand items. A very similar model is proposed in Miranda and Garrido (2004). A similar work that studies inventory issues in multi-location EOQ framework is Schwarz (1981). Shen et al. (2003) develop a column generation-based scheme to solve the same model.

In contrast to the important contributions discussed above, our models try to achieve a system-level service by allocating it to multiple facilities in an optimal manner, i.e., considering the varying fill rates as explicit decision variables. It is well known that it is beneficial to adjust the stock levels and fill rates depending on several properties such as the costs of the parts and their demands (Johnson and Montgomery 1974, Graves et al. 1993, Hopp and Spearman 2000). In this regard, the studies that consider fill rate-based service allocation are closely related to our work (Cohen et al. 1989, 1992).

Part commonality in inventory theory is widely studied and its explicit consideration is known to reduce stock levels saving inventory costs. Collier (1981, 1982), Baker (1985), Baker et al. (1986) and Gerchak et al. (1988) are early studies in this regard. More recently, Thonemann and Brandeau (2000) and Mirchandani

and Mishra (2002) study part commonality in the component design and to satisfy service level constraints, respectively.

1.4 Outline

This dissertation is divided into six chapters and an appendix. In the next chapter, we give a fill rate approximation approach to model and solve the integrated problem. We also include a detailed computational analysis of the issue of integration. In chapter three, we discuss a better approximation approach based on outer-approximation scheme that exploits special structure of the problem for a special case of fill rate calculation, i.e., lost sales setting. Using the same settings of chapter three, in chapter four, we give modeling and optimization techniques for the integrated problem with part commonality. In chapter five, we give further insights by putting the problem in a broader perspective and outline models for more advanced cases and some simplifying special cases. Finally, in chapter six, we conclude the overall research and give various ideas and directions for potential future research.

Chapter 2

Fill Rate Approximation

In this chapter, we present an integrated model based on piece-wise linear approximation of nonlinear fill rate functions. This is a generic approach to model an integrated problem of network design and inventory stocking for any definition of fill rates (i.e., the integrated model can handle fill rates based on both backorders and lost sales). The overall illustration is based on the lost-sales setting.

2.1 Integration Modeling

The issue of integration arises because the service level requirements are imposed on SPL networks. The mathematical models to optimally design and operate SPL networks typically have service level constraints that are based on facility locations being close to customers and on part availability, which is typically measured by fill rates. The fill rates – even modeled as parameters – are the main interface between location/allocation decisions and the inventory decisions. Our modeling effort here considers fill rates as decision variables as functions of location/allocation and stocking level decisions to achieve a system-wide target time-based service level.

2.1.1 Notation

We use the notation in Table 2.1 for modeling.

Table 2.1: Sets/Indices, Parameters and Variables

I	set of candidate facility locations, indexed by i
J	set of customer/demand points, indexed by j
P	set of products, indexed by p
C	set of parts (components), indexed by c
L	set of possible (base) stock levels, indexed by l (see also below for $s_{\max}$)
f_i	fixed annual cost of operating facility i
t_{cij}	cost of shipping a unit of part c from facility i to customer j
d_{cjp}	mean annual demand of part c for product p from customer j
h_{ci}	annual inventory holding cost for part c at facility i
τ_{ci}	replenishment lead time for facility i for part c (in years)
α_p	time-based service level required for product p
δ_{ij}	binary parameter that is 1 if customer j is reachable from facility i within an specified time-window (e.g., 2 hours), 0 otherwise
$s_{\max}$	maximum possible stock level, i.e., $L = \{0, 1, \ldots, s_{\max}\}$
β_{ci}	fill rate of part c at facility i
λ_{ci}	mean lead time demand of part c at facility i
S_{ci}	base stock level of part c at facility i
X_{cijp}	binary decision variable that is 1 if demand for part c for product p from customer j is assigned to facility i, 0 otherwise
Y_i	binary decision variable that is 1 if facility i is open, 0 otherwise

2.1.2 Fill Rate

Fill rate is a term used to determine whether demand for a specific part at a specific location is fulfilled or not. Fill rate, defined as the percentage of total demand for a part satisfied directly from on-hand inventory at a stocking location, is typically used to measure the quality of service at the location. Once an inventory

system is operating with a stocking policy, one could observe the system's response to demand, measure the performance and compute actual (achieved) fill rates, by dividing the amount of demand satisfied directly from stock by the total attempted demand, over a period of time. Actual fill rates computed over short periods of times may fluctuate due to randomness in demand. However, as we lengthen the period over which the fill rate is measured, the actual fill rate stabilizes, especially if the system experiences a stable demand process over time. As the steady-state (long-term) fill rate is a function of the inventory policy, its stock level, and the demand process, one needs to compute the fill rate a priori to be able to design the system, set the stock levels, etc. at the time of planning and decision making, so that the actual fill rates of the designed system are satisfactory. In other words, computing long term fill rates as functions of stock levels and demand process a priori is necessary for planning purposes. Due to stochasticity inherent in these systems, fill rates are computed for steady state behavior of such systems. Numerous mathematical approaches have been developed to analyze steady-state fill rates (Nahmias 1981, Srinivasan et al. 1992, Anupindi and Tayur 1998, Hausman et al. 1998, Cheung and Hausman 1995, Song 1998b). The steady state fill rates as analytical functions of system parameters (probability distribution of stochastic demand, stock level, lead time demand, and the type of the inventory policy) are often complex nonlinear functions derived from mathematical analysis. Two main types of such fill rate functions are defined based on what happens in the real system to the demand not satisfied directly from on-hand stock at the time of attempt. First type is backorder setting, when all the unsatisfied demand is eventually fulfilled by replenishment orders that arrive to the facility that is supposed to serve the original demand. In this case, unsatisfied customer demands are accumulated as backorders

and satisfied as soon as replenishment orders for the facility raise the facility's stock level to satisfactory levels. The second type of fill rate is the case for the lost sales setting, when all the unsatisfied demand is "lost" or gone somewhere outside the system. In this case, the customer demands do not wait for replenishment orders and when a replenishment order arrives, the overall order quantity becomes available for new demands. While this distinction may not seem critical at the outset, yet it is an important one not only because the stochastic process that governs the behavior of the system for each setting is quite different but also because the modeling and solution methodology very much depends on the functional properties of these formulas. There are examples of both backorder and lost sales settings in real SPL systems, but our main focus will be on lost-sales setting. In this chapter, we use the piece-wise linear approximation of the lost-sales fill rate function to handle its nonlinearity. This approach would equally work for the backorder setting with appropriate approximation performed on the backorder fill rate function.

2.1.3 Assumptions

We make the following assumptions for model development:

1. Facilities use *one-for-one replenishment* (also called *base-stock*) policy for inventory replenishment. This is because SPL systems have distinctly low-demand and short lead times hence there is no incentive to order in large batches, especially considering the low ordering costs due to bundled outsourced transportation services. Replenishments to the facilities are provided by a central warehouse with infinite stock level and once ordered, a replenishment from the central warehouse takes a known fixed lead time to arrive at the requesting facility.

2. Demand from each customer location follows a Poisson distribution, which is a very accurate approximation for low-demand systems such as SPL. Two major references in this area use Poisson distribution as to model demand in service parts inventory systems Sherbrooke (1992), Muckstadt (2005). Poisson distribution is also useful because it is completely specified by a single parameter and does not require any variability information (Hopp and Spearman 2000).

3. The demand not satisfied directly from a facility's on-hand stock is passed to an upper-echelon facility (e.g., the central warehouse), which directly ships the part to the requesting customer.

4. Facility fill rates are computed using the lost sales case formula. This is because of the previous assumption.

5. Demand from a customer for a part is assigned to one of the open facilities, but different parts' requests are allowed to come from multiple facilities. This is common in the SPL practice in that one of the facilities serves all of the demand for a part at a customer.

6. A part's demand can be assigned to a facility only if that facility keeps a positive stock level of that part.

7. Only one service time-window is considered for the time-based service level requirement. Although typical SPL systems have multiple service time-windows (with increasing service level requirements for longer time-windows), usually one of the time-windows is the most restrictive, hence assuming one time-window should not hinder the model's value.

8. For the time-window considered, there is one system-wide service level that needs to be satisfied. This system-wide requirement combines all facilities

and customers' demands, and seeks to satisfy a certain percentage of the total system demand to be fulfilled within the specified service time-window. This requirement is a reflection of individual customer service contracts, especially the ones that stipulate an overall service to be achieved across multiple customer locations owned by large customers.

9. Knowledge about which customers a facility can serve within the service time-window is available. As this is usually a function of distance and the mode of transportation available to the facility and customer, we assume that this processing of transportation times is carried out for each pair of customer and facility prior to optimization.

10. The direct shipments from the central warehouse to individual customers in case of stock-outs at the open facilities are always out of the service time-window for all customers.

The first three assumptions are common in SPL literature (Sherbrooke 1992, Muckstadt 2005).

2.1.4 Fill Rate: Lost Sales Case

The fill rate β_{ci} is a function of the base stock level S_{ci} and the mean of the lead time demand λ_{ci} for part c at facility i. The β_{ci} is computed using the lost sales formula (Zipkin 2000):

$$\beta_{ci}(S_{ci}, \lambda_{ci}) = 1 - \frac{\lambda_{ci}^{S_{ci}}/S_{ci}!}{\sum_{n=0}^{S_{ci}} \lambda_{ci}^{n}/n!} \;\forall\; c \in C \text{ and } i \in I. \tag{2.1}$$

This formula is derived from the steady state behavior of $M/G/s/s$ queuing system in which there are s servers and maximum allowed length of queue is also s and the

arrival process is Poisson. We define:

$$\lambda_{ci} = \tau_{ci} \sum_{j \in J} \sum_{p \in P} d_{cjp} X_{cijp} \; \forall \; c \in C \text{ and } i \in I. \tag{2.2}$$

Here, τ_{ci} is the lead time for part c at facility i. Note that the way λ_{ci} values are computed using (2.2) takes care of part commonalities across products since the demand for common parts across products is aggregated here and subsequently there is common stock S_{ci} for part c used in all products.

2.1.5 Service Level Constraints

Using the fill rate values β_{ci} and the demand assignment variables X_{cijp} we write the time-based service level constraints as follows:

$$\sum_{c \in C} \sum_{i \in I} \sum_{j \in J} \delta_{ij} d_{cjp} \beta_{ci} X_{cijp} \geq \alpha_p \sum_{c \in C} \sum_{j \in J} d_{cjp} \; \forall \; p \in P \tag{2.3}$$

The constraints (2.3) state that, for each product p, the fraction of the total system demand for services aggregated across all parts and customers satisfied within the specified time-window must be greater than or equal to the target fraction, α_p. Using (2.1)-(2.3), we present a rather basic model of the problem. This model completely captures the definition of the problem. The basic model (BM) draws heavily from the uncapacitated facility location (UFL) problem and integrates inventory aspects of the problem (costs, fill rates, and service levels):

$$\text{BM: Minimize} \quad \sum_{i\in I} f_i Y_i + \sum_{c\in C}\sum_{i\in I}\sum_{j\in J}\sum_{p\in P} t_{cij} d_{cjp} X_{cijp} + \sum_{c\in C}\sum_{i\in I} h_{ci} S_{ci} \tag{2.4}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{i\in I} X_{cijp} &= 1 \ \forall\ c \in C,\ j \in J, \text{ and } p \in P && (2.5)\\
X_{cijp} &\leq Y_i \ \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P && (2.6)\\
\sum_{c\in C}\sum_{i\in I}\sum_{j\in J} \delta_{ij} d_{cjp} \beta_{ci} X_{cijp} &\geq \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \ \forall\ p \in P && (2.7)\\
\beta_{ci} &= 1 - \frac{\lambda_{ci}^{S_{ci}}/S_{ci}!}{\sum_{n=0}^{S_{ci}} \lambda_{ci}^{n}/n!} \ \forall\ c \in C \text{ and } i \in I && (2.8)\\
\lambda_{ci} &= \tau_{ci} \sum_{j\in J}\sum_{p\in P} d_{cjp} X_{cijp} \ \forall\ c \in C \text{ and } i \in I && (2.9)\\
\beta_{ci}, \lambda_{ci} &\geq 0 \ \forall\ c \in C \text{ and } i \in I && (2.10)\\
S_{ci} &\geq X_{cijp} \ \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P && (2.11)\\
S_{ci} &\geq 0 \text{ and } \textit{integer} \ \forall\ c \in C \text{ and } i \in I && (2.12)\\
X_{cijp}, Y_i &\in \{0,1\} \ \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P. && (2.13)
\end{aligned}$$

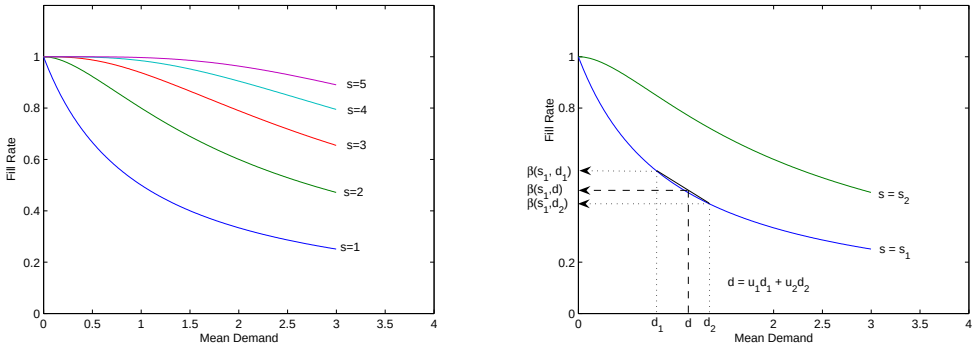
Objective function (2.4) is the total annual costs of opening and operating facilities, delivering parts to customers, and stocking parts at facilities. Constraints (2.5) and (2.6) are standard UFL constraints. Constraints (2.11) state that facilities must maintain positive stock levels if a demand is allocated to them. Rest of the constraints are as explained before. Note that not only the fill rates' definition constraints are nonlinear in decision variables, they are also multiplied by demand assignment decision variables (X_{cijp}) in constraints (2.7). Hence, there is almost no hope to solve BM directly. To be able to solve BM, we use a piece-wise linear approximation scheme for nonlinear fill rate functions to make it amenable to direct optimization solvers.

2.1.6 Fill Rate Approximation

Figure 2.1(a) plots fill rate values for several stock levels as functions of the mean lead time demand. It is apparent that the these functions are rather linear in portions of the curve that correspond to certain ranges of demand. Using this observation, we approximate the fill rate curves with piece-wise linear approximation. For each part c, we divide the possible range (0 to $\mu_c^{\max}$) of demand axis in several intervals using μ_{bc} values computed on a set of break points B (indexed by b). To compute the range, we calculate $\mu_c^{\max}$ as the maximum amount of demand that can be assigned to a facility. We define ρ_{bcl} as the fill rate value computed using (2.1) by setting $S_{ci} = l$ and $\lambda_{ci} = \mu_{bc}$ for all i. For a given set of demand break points μ_{bc} for all $b \in B$, and stock levels $l \in L$, we can construct a table of fill rate values for each part c. An illustrative fill rate table is shown in Table 2.2 for stock levels up to $s_{\max} = 5$ and a maximum mean lead time demand $\mu_c^{\max} = 3$. To calculate the fill rates for any value of λ_{ci}, we use table look-up and linear interpolation with pre-computed fill rate values. This process is illustrated in Figure 2.1(b). If λ_{ci} is in between $\mu_{b-1,c}$ and μ_{bc} then we can write $\lambda_{ci} = u_1\mu_{b-1,c} + u_2\mu_{bc}$, where $u_1 + u_2 = 1$). Using u_1 and u_2, we approximate fill rate between the two pre-computed fill rate values $\rho_{b-1,c,l}$ and ρ_{bcl} as $\beta(l, \lambda_{ci}) = u_1\rho_{b-1,c,l} + u_2\rho_{bcl}$ for each stock level l.

Table 2.2: An example fill rate table

μ_{bc}	(0)	(μ_{1l})	(μ_{2l})	(μ_{3l})	(μ_{4l})	(μ_{5l})	(μ_{6l})	$(\mu_{7l} = s^{\max})$
$b \in B$	0.0	0.3	0.6	0.9	1.2	1.8	2.4	3.0
s = 1	1.000	0.769	0.625	0.526	0.455	0.357	0.294	0.250
s = 2	1.000	0.967	0.899	0.824	0.753	0.633	0.541	0.471
s = 3	1.000	0.997	0.980	0.950	0.910	0.820	0.732	0.654
s = 4	1.000	1.000	0.997	0.989	0.974	0.925	0.861	0.794
s = 5	1.000	1.000	1.000	0.998	0.994	0.974	0.938	0.890



(a) Fill Rates

(b) Linear Interpolation

Figure 2.1: Fill rates and linear interpolation $\beta(s_1, d) = u_1\beta(s_1, d_1) + u_2\beta(s_1, d_2)$ with $u_1 + u_2 = 1$

2.1.7 The Linear Model

To correctly identify the demand interval in which a λ_{ci} lies, we define new type 2 SOS (special ordered set) variables U_{bci} (representing u's in the above example) for interpolation. We also define binary variables V_{cil} that take value 1 if S_{ci} must be l, and 0 otherwise. With the addition of these variables, we present the linear model (LM):

$$\text{LM: Minimize} \quad \sum_{i\in I} f_i Y_i + \sum_{c\in C}\sum_{i\in I}\sum_{j\in J}\sum_{p\in P} t_{cij} d_{cjp} X_{cijp} + \sum_{c\in C}\sum_{i\in I} h_{ci} S_{ci} \tag{2.14}$$

$$\text{s.t.} \quad \sum_{i\in I} X_{cijp} = 1 \;\forall\; c \in C,\; j \in J, \text{ and } p \in P \tag{2.15}$$

$$X_{cijp} \leq Y_i \;\forall\; c \in C,\; i \in I,\; j \in J, \text{ and } p \in P \tag{2.16}$$

$$\sum_{c\in C}\sum_{i\in I}\sum_{j\in J} \delta_{ij} d_{cjp} \beta_{ci} X_{cijp} \geq \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \;\forall\; p \in P \tag{2.17}$$

$$\lambda_{ci} = \tau_{ci} \sum_{j\in J}\sum_{p\in P} d_{cjp} X_{cijp} \;\forall\; c \in C \text{ and } i \in I \tag{2.18}$$

$$\lambda_{ci} \leq \sum_{b \in B} \mu_{bc} U_{bci} \ \forall\ c \in C \text{ and } i \in I \quad (2.19)$$

$$S_{ci} \geq \sum_{l \in L} l V_{cil} \ \forall\ c \in C \text{ and } i \in I \quad (2.20)$$

$$S_{ci} \geq X_{cijp} \ \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P \quad (2.21)$$

$$\beta_{ci} \leq \sum_{l \in L} \sum_{b \in B} \rho_{bcl} U_{bci} V_{cil} \ \forall\ c \in C \text{ and } i \in I \quad (2.22)$$

$$\sum_{b \in B} U_{bci} = 1 \ \forall\ c \in C \text{ and } i \in I \quad (2.23)$$

$$\sum_{l \in L} V_{cil} = 1 \ \forall\ c \in C \text{ and } i \in I \quad (2.24)$$

$$\beta_{ci}, \lambda_{ci}, S_{ci} \geq 0 \ \forall\ c \in C \text{ and } i \in I \quad (2.25)$$

$$V_{cil}, X_{cijp}, Y_i \in \{0,1\} \ \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P \quad (2.26)$$

$$U_{bci} \in [0,1] \text{ and } SOS2 \ \forall\ b \in B,\ c \in C, \text{ and } i \in I. \quad (2.27)$$

In LM, the objective function and first four sets of constraints are exactly the same as in BM. Constraints (2.19)-(2.24) represent process of linear interpolation of fill rates and identification of stock level for each S_{ci} variable. Constraints (2.19) and (2.23) locate λ_{ci} values in between two consecutive demand breakpoints in the pre-computed fill rate table for part c. Consecutiveness of nonzero U_{bci} variables is taken care by their $SOS2$ declaration (Kalvalagen 2002). Constraints (2.20) and (2.24) select the stock level of part c at facility i such that the required fill rate to satisfy service levels is available. The fill rates are calculated using constraints (2.22), which makes use of the stock selection variables V_{cil} and demand identification variables U_{bci}. The nonlinearities in (2.17) and (2.22) are easier to handle, we replace the nonlinear terms $(\beta_{ci} X_{cijp})$ and $(U_{bci} V_{cil})$ by two new variables, $Z_{cijp} \in [0,1]$ and $W_{bcil} \in [0,1]$, respectively. The replacement constraints $W_{bcil} = U_{bci} V_{cil}, \forall\ b \in B,\ c \in C,\ i \in I, l \in L$, and $Z_{cijp} = \beta_{ci} X_{cijp}, \forall\ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P$ can

be linearized. We write the following new constraints to replace (2.17) and (2.22):

$$\sum_{c\in C}\sum_{i\in I}\sum_{j\in J}\delta_{ij}d_{cjp}Z_{cijp} \geq \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \;\forall\; p \in P \tag{2.28}$$

$$\beta_{ci} \leq \sum_{l\in L}\sum_{b\in B}\rho_{bcl}W_{bcil} \;\forall\; c \in C \text{ and } i \in I \tag{2.29}$$

$$W_{bcil} \geq U_{bci} + V_{cil} - 1 \tag{2.30}$$

$$W_{bcil} \leq U_{bci} - V_{cil} + 1 \tag{2.31}$$

$$W_{bcil} \leq U_{bci} \tag{2.32}$$

$$W_{bcil} \leq V_{cil} \tag{2.33}$$

$$Z_{cijp} \geq \beta_{ci} + X_{cijp} - 1 \tag{2.34}$$

$$Z_{cijp} \leq \beta_{ci} - X_{cijp} + 1 \tag{2.35}$$

$$Z_{cijp} \leq \beta_{ci} \tag{2.36}$$

$$Z_{cijp} \leq X_{cijp}. \tag{2.37}$$

2.2 Two-Stage Modeling

The two-stage modeling approach decomposes the problem by decoupling the network design and inventory decisions. The first stage is modeled assuming that the fixed and free fill rates are available at all facilities while the second stage computes the costs of fill rates and associated stock levels once location/allocation decisions are made in the first stage. This way, the first stage model is a modified uncapacitated facility location model (LOC) with service constraints, and the second stage model is a multi-part, multi-facility, service-constrained inventory stocking (INV) model.

2.2.1 The Location/Allocation Model

$$\text{LOC: Minimize} \quad \sum_{i\in I} f_i Y_i + \sum_{c\in C}\sum_{i\in I}\sum_{j\in J}\sum_{p\in P} t_{cij} d_{cjp} X_{cijp} \tag{2.38}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{i\in I} X_{cijp} &= 1 \ \forall\ c\in C,\ j\in J, \text{ and } p\in P && (2.39)\\
X_{cijp} &\le Y_i \ \forall\ c\in C,\ i\in I,\ j\in J, \text{ and } p\in P && (2.40)\\
\sum_{c\in C}\sum_{i\in I}\sum_{j\in J} \delta_{ij} d_{cjp} X_{cijp} &\ge \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \ \forall\ p\in P && (2.41)\\
X_{cijp}, Y_i &\in \{0,1\} \ \forall\ c\in C,\ i\in I,\ j\in J, \text{ and } p\in P. && (2.42)
\end{aligned}$$

This model is rather self-explanatory. The main change from a multi-commodity version of the UFL model is the addition of service level constraints (2.41) with free and 100% fill rates. The service level constraints here are just demand-coverage constraints. We denote $\hat{Y}_i$ and $\hat{X}_{cijp}$ the optimal values of corresponding variables on solving LOC. We define $\hat{I} = \{i \in I : \hat{Y}_i = 1\}$ and determine $\hat{U}_{bci}$ and $\hat{\lambda}_{ci}$ values using the following relations:

$$\begin{aligned}
\hat{\lambda}_{ci} &= \tau_{ci} \sum_{j\in J}\sum_{p\in P} d_{cjp}\hat{X}_{cijp} \ \forall\ c\in C \text{ and } i\in \hat{I}\\
\hat{\lambda}_{ci} &\le \sum_{b\in B} U_{bci}\mu_{bc} \ \forall\ c\in C \text{ and } i\in \hat{I}.
\end{aligned}$$

We iterate over $b \in B$ and find the smallest b, say $\hat{b}$, such that $\mu_{\hat{b}c}$ is greater than or equal to $\hat{\lambda}_{ci}$, and set

$$\hat{U}_{\hat{b}-1,c,i} = \frac{\mu_{\hat{b}c} - \hat{\lambda}_{\hat{b}ci}}{\mu_{\hat{b}c} - \mu_{\hat{b}-1,c}} \quad \text{and} \quad \hat{U}_{\hat{b}ci} = 1 - \hat{U}_{\hat{b}-1,c,i}$$
$$\hat{U}_{bci} = 0,\ \forall\ b\in B, b\neq \hat{b}, b\neq \hat{b}-1.$$

2.2.2 The Inventory Model

Using LM and $\hat{X}_{cijp}, \hat{Y}_i, \hat{\lambda}_{ci}$ and $\hat{U}_{bci}$ as fixed values of corresponding variables, we write the multi-facility inventory stocking model:

$$\text{INV: Minimize} \quad \sum_{c\in C}\sum_{i\in \hat{I}} h_{ci}S_{ci} \tag{2.43}$$

$$\begin{aligned}
\text{s.t. } S_{ci} &\geq \sum_{l\in L} lV_{cil} \;\forall\; c\in C \text{ and } i\in \hat{I} && (2.44)\\
\sum_{l\in L} V_{cil} &= 1 \;\forall\; c\in C \text{ and } i\in \hat{I} && (2.45)\\
\sum_{c\in C}\sum_{i\in \hat{I}}\sum_{j\in J} \delta_{ij}d_{cjp}\beta_{ci}\hat{X}_{cijp} &\geq \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \;\forall\; p\in P && (2.46)\\
\beta_{ci} &\leq \sum_{l\in L}\sum_{b\in B} \rho_{bcl}\hat{U}_{bci}V_{cil} \;\forall\; c\in C \text{ and } i\in \hat{I} && (2.47)\\
V_{cil} &\in \{0,1\} \;\forall\; c\in C,\; i\in \hat{I},\; \text{and } l\in L && (2.48)\\
\beta_{ci}, S_{ci} &\geq 0 \;\forall\; c\in C \text{ and } i\in \hat{I}. && (2.49)
\end{aligned}$$

The main decision variables in INV are S_{ci} variables that affect the fill rates, which eventually determine which combinations of stock levels are needed to satisfy the service levels constraints.

The solution of the two-stage approach is the combined solution of LOC and INV with the total cost as the summation of the objectives of the two models. The two-stage modeling and solution technique is quite efficient as both of the models are linear, smaller, and often amenable to direct optimization. However, the technique has its disadvantages in terms of cost, solution quality, and feasibility. The first stage assumes 100% fill rates, which if necessary (to obtain to satisfy the service level constraints) may prove costly or even cause infeasibility in the second stage.

This way of decoupling the two sets of decision may not be a wise thing to do, however, when solving an integrated model like LM is a challenge, we can obtain potentially good upper bounding solutions using two-stage solution. We can also modify LOC and INV using Lagrangian decomposition and solve them in sequence iteratively to link the two sets of decisions.

2.2.3 Lagrangian Decomposition

Lagrangian decomposition (Guignard and Kim 1987) of LM gives us a two-stage solution scheme to compute lower and upper bounds. The decomposition breaks the LM model into two smaller MIP models. To illustrate the decomposition, we define the following two copy variables:

$$\chi_{cijp} = X_{cijp} \; \forall \; c \in C, \; i \in I, \; j \in J, \text{ and } p \in P \tag{2.50}$$

$$\eta_{ci} = \beta_{ci} \; \forall \; c \in C, \text{ and } i \in I. \tag{2.51}$$

Using the copy variables we can modify LM by replacing X_{cijp} with χ_{cijp} in (2.18) and replacing β_{ci} with η_{ci} in (2.22). Then using Lagrangian relaxation of (2.50) and (2.51) with u_{cijp} and v_{ci} as Lagrange multipliers respectively, we can separate the objective function terms and constraints to write modified LOC and INV as MLOC and MINV respectively. MLOC has now cost terms for fill rates and handles the location, allocation and fill rate variables without the explicit consideration of mean lead time demands and stock levels:

$$\text{MLOC: Minimize} \quad \sum_{i \in I} f_i Y_i + \sum_{c \in C} \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} (t_{cij} d_{cjp} - u_{cijp}) X_{cijp} + \sum_{c \in C} \sum_{i \in I} v_{ci} \beta_{ci} \tag{2.52}$$

$$\text{s.t.} \quad \sum_{i \in I} X_{cijp} = 1 \ \forall \ c \in C,\ j \in J, \text{ and } p \in P \tag{2.53}$$

$$X_{cijp} \leq Y_i \ \forall \ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P \tag{2.54}$$

$$\sum_{c \in C} \sum_{i \in I} \sum_{j \in J} \delta_{ij} d_{cjp} \beta_{ci} X_{cijp} \geq \alpha_p \sum_{c \in C} \sum_{j \in J} d_{cjp} \ \forall \ p \in P \tag{2.55}$$

$$X_{cijp}, Y_i \in \{0,1\} \ \forall \ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P \tag{2.56}$$

$$0 \leq \beta_{ci} \leq 1 \ \forall \ c \in C \text{ and } i \in I. \tag{2.57}$$

MINV mainly handles the demand allocations, mean lead time demands, fill rates, along with the explicitly considered stock levels:

$$\text{MINV: Minimize} \quad \sum_{c \in C} \sum_{i \in I} (h_{ci} S_{ci} - v_{ci} \eta_{ci}) + \sum_{c \in C} \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} u_{cijp} \chi_{cijp} \tag{2.58}$$

$$\text{s.t.} \quad \sum_{i \in I} \chi_{cijp} = 1 \ \forall \ c \in C,\ j \in J, \text{ and } p \in P \tag{2.59}$$

$$\sum_{c \in C} \sum_{i \in I} \sum_{j \in J} \delta_{ij} d_{cjp} \eta_{ci} \chi_{cijp} \geq \alpha_p \sum_{c \in C} \sum_{j \in J} d_{cjp} \ \forall \ p \in P \tag{2.60}$$

$$\lambda_{ci} = \tau_{ci} \sum_{j \in J} \sum_{p \in P} d_{cjp} \chi_{cijp} \ \forall \ c \in C \text{ and } i \in I \tag{2.61}$$

$$\lambda_{ci} \leq \sum_{b \in B} \mu_{bc} U_{bci} \ \forall \ c \in C \text{ and } i \in I \tag{2.62}$$

$$S_{ci} \geq \sum_{l \in L} l V_{cil} \ \forall \ c \in C \text{ and } i \in I \tag{2.63}$$

$$\eta_{ci} \leq \sum_{l \in L} \sum_{b \in B} \rho_{bcl} U_{bci} V_{cil} \ \forall \ c \in C \text{ and } i \in I \tag{2.64}$$

$$\sum_{b \in B} U_{bci} = 1 \ \forall \ c \in C \text{ and } i \in I \tag{2.65}$$

$$\sum_{l \in L} V_{cil} = 1 \ \forall \ c \in C \text{ and } i \in I \tag{2.66}$$

$$U_{bci} \in [0,1] \text{ and } SOS2 \ \forall \ b \in B,\ c \in C, \text{ and } i \in I \tag{2.67}$$

$$V_{cil}, \chi_{cijp} \in \{0,1\} \ \forall \ c \in C,\ i \in I,\ j \in J, \text{ and } p \in P \tag{2.68}$$

$$\eta_{ci}, \lambda_{ci}, S_{ci} \geq 0 \ \forall \ c \in C \text{ and } i \in I. \tag{2.69}$$

For given values of u_{cijp} and v_{ci}, the two models when solved to optimality independently give us a lower bound on the objective function value of LM. The lower bound is obtained when the two optimal objective function values are added. An upper bounding heuristic can also be developed if we solve the two models in tandem, i.e., we solve the MLOC first and fix the values of χ_{cijp} variables in MINV to the optimal values of X_{cijp} in MLOC and then solve MINV. The combined solution thus obtained, if feasible to LM, will give an upper bound on the objective function value of LM. We may not obtain a feasible solution for every given set of values for u_{cijp} and v_{ci}, however, we can use subgradient optimization to update u_{cijp} and v_{ci} values and obtain multiple lower and upper bounds and keep the best. The two models MLOC and MINV are different from LOC and INV largely in objective function terms. The additional terms (penalty terms) link the two models and provide a feedback mechanism for successive correction and convergence. The nonlinearity in constraints (2.55), (2.60), and (2.64) can be handled same as before, i.e., by replacing the product of two variables by a single variable.

2.3 The Linear Model: Ignore Part Commonality

When part commonality is ignored and separate stocks of a part are maintained to service different products, we need to differentiate the demands, stocks, and fill rates across products. In our mathematical model IGN-LM we use a superscript "p" on λ_{ci} , U_{bci}, V_{cil}, β_{ic}, and S_{ci} variables, and write them as λ^p_{ci} , U^p_{bci}, V^p_{cil}, β^p_{ic}, and S^p_{ci}. In this case, for example, β^p_{ic} denotes the fill rate of part c to be used for product p demand at facility i. Using this notation, we present a linear model (IGN-LM):

$$\text{IGN-LM: Minimize } \sum_{i\in I} f_i Y_i + \sum_{c\in C}\sum_{i\in I}\sum_{j\in J}\sum_{p\in P} t_{cij} d_{cjp} X_{cijp} + \sum_{c\in C}\sum_{i\in I}\sum_{p\in P} h_{ci} S^p_{ci} \quad (2.70)$$

$$\begin{aligned}
\text{s.t. } \sum_{i\in I} X_{cijp} &= 1 \ \forall\ c\in C,\ j\in J, \text{ and } p\in P && (2.71)\\
X_{cijp} &\le Y_i \ \forall\ c\in C,\ i\in I,\ j\in J, \text{ and } p\in P && (2.72)\\
\sum_{c\in C}\sum_{i\in I}\sum_{j\in J} \delta_{ij} d_{cjp} \beta^p_{ci} X_{cijp} &\ge \alpha_p \sum_{c\in C}\sum_{j\in J} d_{cjp} \ \forall\ p\in P && (2.73)\\
\lambda^p_{ci} &= \tau_{ci} \sum_{j\in J} d_{cjp} X_{cijp} \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.74)\\
\lambda^p_{ci} &\le \sum_{b\in B} \mu_{bc} U^p_{bci} \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.75)\\
S^p_{ci} &\ge \sum_{l\in L} l V^p_{cil} \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.76)\\
S^p_{ci} &\ge X_{cijp} \ \forall\ c\in C,\ i\in I,\ j\in J, \text{ and } p\in P && (2.77)\\
\beta^p_{ci} &\le \sum_{l\in L}\sum_{b\in B} \rho_{bcl} U^p_{bci} V^p_{cil} \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.78)\\
\sum_{b\in B} U^p_{bci} &= 1 \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.79)\\
\sum_{l\in L} V^p_{cil} &= 1 \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.80)\\
\beta^p_{ci}, \lambda^p_{ci}, S^p_{ci} &\ge 0 \ \forall\ c\in C,\ i\in I, \text{ and } p\in P && (2.81)\\
V^p_{cil}, X_{cijp}, Y_i &\in \{0,1\} \ \forall\ c\in C,\ i\in I,\ j\in J, \text{ and } p\in P && (2.82)\\
SOS2\ \ U^p_{bci} &\in [0,1] \ \forall\ b\in B,\ c\in C,\ i\in I, \text{ and } p\in P. && (2.83)
\end{aligned}$$

IGN-LM is very similar to LM except that there is no interaction in the demand and stock levels across products. In fact, the first term in the objective function ($\sum_{i\in I} f_i Y_i$) is the only term that is shared across products. If that term is ignored (e.g., if it is possible $f_i = 0\ \forall\ i \in I$), then IGN-LM can be decomposed into

subproblems across products and each subproblem can be solved independently.

As a way of obtaining decoupled solutions to the ignore part commonality setting, we can develop two-stage models, similar to the ones in Section 2.2, except that both now ignore part commonality.

2.4 Computational Study

In this section, we do a preliminary computation using the models developed in the previous sections. We present results using four different models which are obtained from the following combinations:

1. Integrated model with part commonality ignored,
2. Integrated model with part commonality exploited,
3. Two-stage model with part commonality ignored, and
4. Two-stage model with part commonality exploited.

2.4.1 Test Problem Data

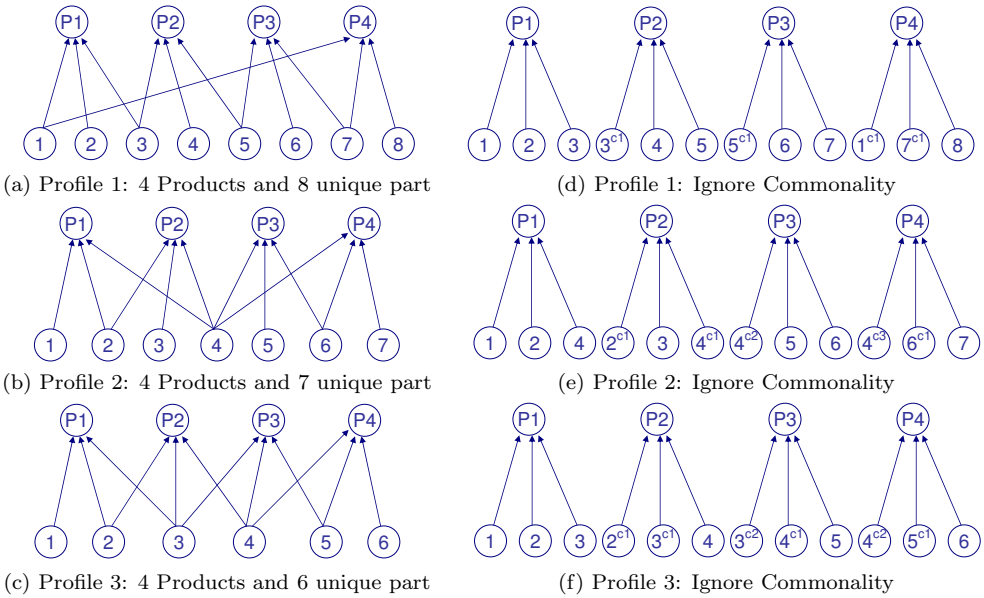
The test problem instances involve 25 customers, 12 candidate facilities, 4 products and 12 parts. Customer and facility locations are generated uniformly on a grid of 1000×1000. The transportation costs t_{cij} are calculated based on the Euclidean distance between i and j. To generate transportation costs, we use a step function to create three distance categories: low-cost shipping distance, medium-cost shipping distance, and high-cost shipping distance. A t_{cij} value in any category is the one tenth of the average Euclidean distance in the category under consideration. The t_{cij} values are rounded to the nearest positive integers. Mean demand values (d_{cjp})

are generated as multiplication of two random numbers, the part failure rate ω_c and the number of products n_{jp} customer j owns. The ω_c values are generated uniformly within the interval $(0.04, 0.1)$ and n_{jp} uniformly within the interval $[3, 7]$ and integer, and finally we set $d_{cjp} = n_{jp} * \omega_c$. Holding costs ($h_{ci}$) across all facilities are generated uniformly within the interval (50, 150), and integer. Assuming the mean demand values are for a year, we use lead times of one week for all facilities and parts. Service time-window indicators δ_{ij} are determined using the 25% of grid size, i.e. $\delta_{ij} = 1$, if the Euclidean distance between i and j is less than 250, 0 otherwise. The maximum stock level $s_{\max}$ is 5. We use 4 demand break points in the fill rate approximation. Each break point corresponds to a percentage of the maximum mean lead time demand ($d_c^{\max}$), which is calculated separately for each part c, as sum of all d_{cjp} values over all customers $j \in J$ and all products $p \in P$. A demand value at the break point (μ_{bc}) are calculated for each part by multiplying the maximum mean lead time demand with the corresponding percentage value:

$$\mu_{bc} = \gamma_b * d_c^{\max} \text{ and } \gamma_b \in (0.25, 0.50, 0.75, 1.00)$$

where γ_b is the percentage corresponding to bth break point. The fill rate table lookup values (ρ_{bcl}) are calculated explicitly using formula (2.1) for each μ_{bc} and l values. We use the same time based service levels α_p for all products, which is varied as an experimental factor, taking one of three values, 40%, 60%, or 80%. Another factor we vary in the experiments is the fixed facility costs f_i, which takes one of three values, 100, 1000 and 10000. We generate part commonality in three profiles (see figure 2.2) with varying number of common parts across a number of products. Corresponding to each commonality profile shown in the left column 2.2(a)-(c), we also show associated "ignore commonality" cases in the right column

2.2(d)-(f). We replicate model data and solutions for three seeds for random number generation (customer and facility locations, their demands and holding costs) for each combination of varying levels of the factors. In the tables, we show results from a representative seed. We use CPLEX 9.0 installed on a PC with dual Xeon 1.8 GHz processors and 1 GB RAM running Suse Linux for all computations. We use a stopping criterion of time limit of 900 seconds and 1% optimality gap (whichever occurs first). In almost all the instances time limit reached and the solution reported are best upper bounding solution obtained by CPLEX.



(a) Profile 1: 4 Products and 8 unique part

(d) Profile 1: Ignore Commonality

(b) Profile 2: 4 Products and 7 unique part

(e) Profile 2: Ignore Commonality

(c) Profile 3: 4 Products and 6 unique part

(f) Profile 3: Ignore Commonality

Figure 2.2: Three different profiles considered for part commonalities.

2.4.2 Results and Discussion

Table 2.3 presents results using the integrated model. This table is divided into four groups of columns: the first group of three columns defines input parameters to the model, other two groups show results for the two cases of part commonality and the

Table 2.3: Computational Results with Integrated Model

Problem			Part Commonality Ignored			Part Commonality Exploited			Comparison
f_i	α_p	Profile†	Cost	n^*	Facilities	Cost	n	Facilities	%Savings‡
		1	2016	3	9,10,12	1,702	2	4,10	18.43
	0.4	2	2044	3	9,10,12	1,516	2	4,10	34.85
		3	2094	3	9,10,12	1,602	2	4,10	30.72
		1	2283	3	4,9,10	1,890	3	4,9,10	20.82
100	0.6	2	2371	4	3,4,9,10	1,753	3	4,9,10	35.26
		3	2404	4	3,4,9,10	1,834	3	4,9,10	31.09
		1	2936	5	1,3,4,9,10	2,295	5	1,3,4,9,10	27.93
	0.8	2	2886	5	1,3,4,9,10	2,162	5	1,3,4,9,10	33.49
		3	2996	5	1,3,4,9,10	2,301	5	1,3,4,9,10	30.18
		1	3867	2	4,10	3,502	2	4,10	10.41
	0.4	2	3888	2	4,10	3,316	2	4,10	17.26
		3	3955	2	4,10	3,402	2	4,10	16.25
		1	5007	3	4,9,10	4,590	3	4,9,10	9.09
1000	0.6	2	5081	3	4,9,10	4,453	3	4,9,10	14.12
		3	5153	3	4,9,10	4,534	3	4,9,10	13.66
		1	7368	5	1,3,4,9,10	6,797	5	1,3,4,9,10	8.40
	0.8	2	7441	5	1,3,4,9,10	6,706	5	1,3,4,9,10	10.96
		3	7562	5	1,3,4,9,10	6,804	5	1,3,4,9,10	11.13
		1	21877	2	4,10	21,648	2	10,12	1.06
	0.4	2	21910	2	4,10	21,368	2	10,12	2.54
		3	22020	2	4,10	21,548	2	10,12	2.19
		1	32145	3	4,9,10	31,719	3	3,4,10	1.34
10000	0.6	2	32165	3	3,4,10	31,510	3	3,4,10	2.08
		3	32312	3	4,9,10	31,534	3	4,9,10	2.47
		1	52539	5	1,3,9,10,12	51,795	5	1,3,4,9,10	1.44
	0.8	2	52608	5	1,3,4,9,10	52,046	5	1,3,4,9,10	1.08
		3	52634	5	1,4,6,9,10	51,844	5	1,3,4,9,10	1.52

† part commonality profile
‡ % savings in overall cost due to part part commonality consideration
* total number of open facilities

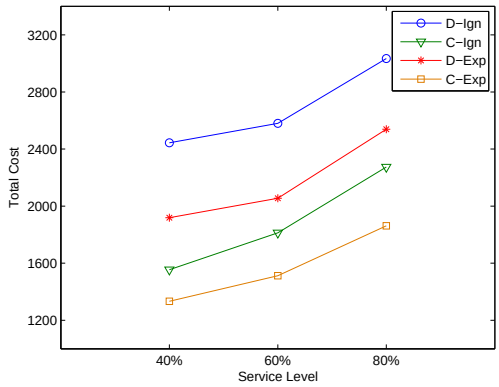
last column shows the percentage savings in the overall cost when part commonality is exploited in network design and inventory stocking decisions. Table 2.4, with the same structure, shows the results for the same set of problem instances using

Table 2.4: Computational Results with Two-stage Model

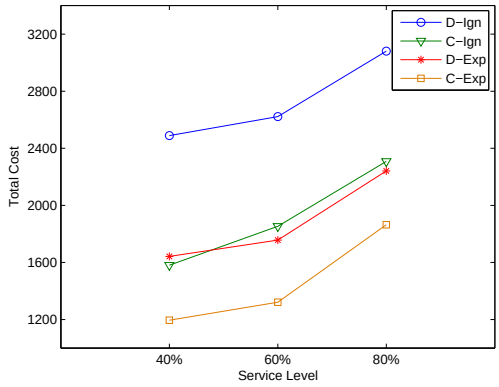
Problem			Part Commonality Ignored			Part Commonality Exploited			Comparison
f_i	α_p	Profile†	Cost	n^*	Facilities	Cost	n	Facilities	%Savings‡
		1	3040	4	3,4,7,9	2,360	4	3,4,7,9	28.81
	0.4	2	3120	4	3,4,7,9	2,260	4	3,4,7,9	38.05
		3	3280	4	3,4,7,9	2,340	4	3,4,7,9	40.17
		1	3067	4	3,4,9,10	2,387	4	3,4,9,10	28.49
100	0.6	2	3147	4	3,4,9,10	2,287	4	3,4,9,10	37.60
		3	3307	4	3,4,9,10	2,367	4	3,4,9,10	39.71
		1	4163	5	1,3,4,9,10	3,383	5	1,3,4,9,10	23.06
	0.8	2	4153	5	1,3,4,9,10	3,083	5	1,3,4,9,10	34.71
		3	4333	5	1,3,4,9,10	3,263	5	1,3,4,9,10	32.79
		1	4228	2	10,12	3,648	2	10,12	15.90
	0.4	2	4228	2	10,12	3,368	2	10,12	25.53
		3	4328	2	10,12	3,548	2	10,12	21.98
		1	5677	3	9,10,12	4,957	3	9,10,12	14.52
1000	0.6	2	5687	3	9,10,12	4,607	3	9,10,12	23.44
		3	5837	3	9,10,12	4,807	3	9,10,12	21.43
		1	8663	4	1,3,4,9,10	7,883	5	1,3,4,9,10	9.89
	0.8	2	8653	4	1,3,4,9,10	7,583	5	1,3,4,9,10	14.11
		3	8833	4	1,3,4,9,10	7,763	5	1,3,4,9,10	13.78
		1	22228	2	10,12	21,648	2	10,12	2.68
	0.4	2	22228	2	10,12	21,368	2	10,12	4.02
		3	22328	2	10,12	21,548	2	10,12	3.62
		1	32677	3	9,10,12	31,957	3	9,10,12	2.25
10000	0.6	2	32687	3	9,10,12	31,607	3	9,10,12	3.42
		3	32837	3	9,10,12	31,807	3	9,10,12	3.24
		1	53663	5	1,3,4,9,10	52,883	5	1,3,4,9,10	1.47
	0.8	2	53653	5	1,3,4,9,10	52,583	5	1,3,4,9,10	2.03
		3	53833	5	1,3,4,9,10	52,763	5	1,3,4,9,10	2.03

† part commonality profile
‡ % savings in overall cost due to part part commonality consideration
* total number of open facilities

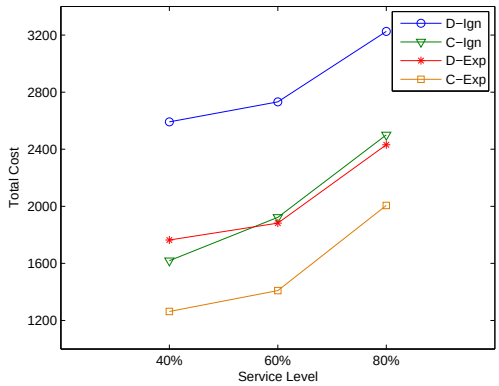
the two-stage approach. We observe that apart from savings in overall cost, part commonalities affect the network design decisions especially the set of open facilities. This fact is revealed when the network design and inventory stocking problems are



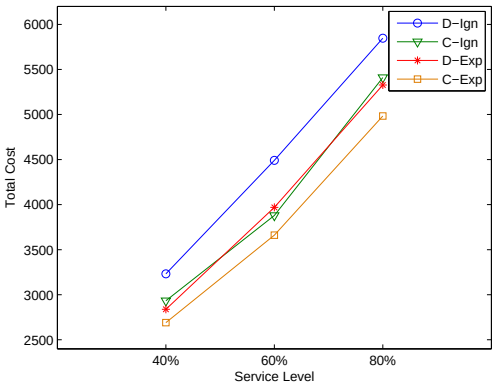
(a) $f_i = 100$ and Profile-1



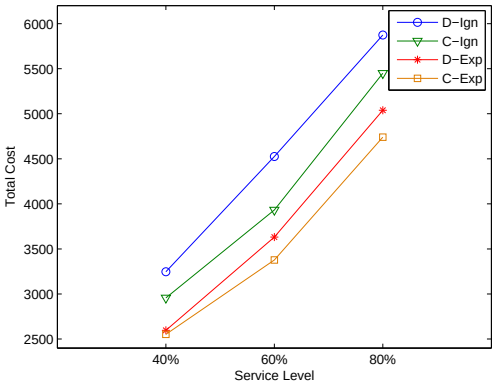
(b) $f_i = 100$ and Profile-2



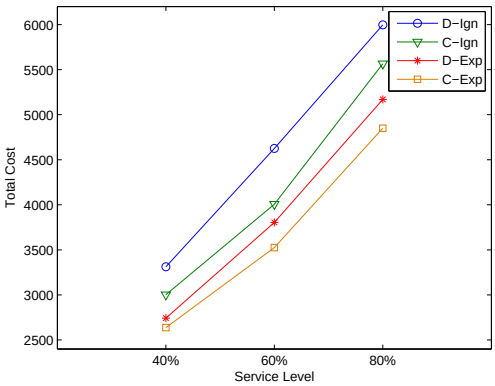
(c) $f_i = 100$ and Profile-3



(d) $f_i = 1000$ and Profile-1



(e) $f_i = 1000$ and Profile-2



(f) $f_i = 1000$ and Profile-3

Figure 2.3: Comparing objective function values for four different models.

solved in the integrated fashion. Figure 2.3 compares the total costs obtained using the four solutions for the two intermediate levels of part commonality (6 and 3 unique parts) showing each level of fixed facility cost in a separate graph. Graphs in the left column, i.e., 2.3(a)-(c), show the tradeoff curves for instances with 6 unique parts, and graphs in the right column 2.3(d)-(f) show them for instances with 3 unique parts. In the graphs, "C" refers to the integrated (Combined) model, "D" to the two-stage (Decoupled) approach, "Ign" refers to ignoring part commonality, and "Exp" to exploiting part commonalities. From these graphs, we see that overall costs increase with increasing the service level requirements. In all cases illustrated in Figure 2.3, "D-Ign" draws the upper envelope, providing the solutions with highest costs, and "C-Exp" draws the lower envelope, which shows potential costs savings through integration and consideration of part commonalities.

2.5 Summary

In this chapter we presented an approach to model and solve the integrated problem of network design and inventory stocking through piece-wise linear approximation of fill rates. The computational results show the merits of integration and effectiveness of overall approach as it also lets us exploit part commonalities. The approximation scheme and development of LM are not limited to any specific formula for fill rate computation. The piece-wise linear approximation scheme, however accurate and effective, may present challenges in solving large MIPs – complexity of which may grow exponentially as the problem size in terms of number of customers, candidate facilities, and parts grows. For handling nonlinearities, piece-wise linear approximation is just one approach among many. More effective approximation schemes must be exploited, as we attempt in the next chapter.

Chapter 3

Variable Substitution and θ Approximation

In this chapter we present a more effective approach to solve the integrated problem when fill rates are computed using a specific formula applicable for the "lost sales case" for the Poisson distributed demand. For the ease of development, we first consider a single part and single product version of the problem. We use the same notation as before but suppress "c" and "p" from subscripts in the entire development.

3.1 Notation

Given a set of stocking locations I (indexed by i) and a set of customer demand points J (indexed by j), we define parameter d_j as the mean annual demand rate at demand point j, and α is the required fraction (service level) of the system-wide total demand to be satisfied within the time-window. The replenishment lead time (in years) is denoted by τ, assumed to be the same for all facilities. Parameters f_i,

t_{ij}, and h_i are annual costs of operating facility i, transporting part from facility i to customer j and inventory holding cost at facility i, respectively. Decision variables $Y_i \; \forall \; i \in I$ are binary, indicating if facility i is open/stocked. Decision variables X_{ij} for all i and j are also binary, taking value 1 when demand point j is assigned to facility i, and 0 otherwise. Nonnegative variables β_i, λ_i, and $S_i \; \forall \; i \in I$ are the fill rate, mean lead time demand, and stock level, respectively, at facility i – same as defined in the previous chapter.

3.2 Modeling

We first present a version of the basic model for the single part and single product case (SBM) below:

$$\text{SBM: Minimize} \quad \sum_{i \in I} f_i Y_i + \sum_{i \in I} \sum_{j \in J} t_{ij} d_j X_{ij} + \sum_{i \in I} h_i S_i \tag{3.1}$$

$$\text{s.t.} \quad \sum_{i \in I} X_{ij} = 1 \; \forall \; j \in J \tag{3.2}$$

$$X_{ij} \leq Y_i \; \forall \; i \in I \text{ and } j \in J \tag{3.3}$$

$$\sum_{i \in I} \sum_{j \in J} \delta_{ij} d_j \beta_i X_{ij} \geq \alpha \sum_{j \in J} d_j \tag{3.4}$$

$$\lambda_i = \tau \sum_{j \in J} d_j X_{ij} \; \forall \; i \in I \tag{3.5}$$

$$\beta_i = 1 - \frac{\lambda^{S_i}/S_i!}{\sum_{n=0}^{S_i} \lambda_i^n/n!} \; \forall \; i \in I \tag{3.6}$$

$$S_i \geq X_{ij} \; \forall \; i \in I \text{ and } j \in J \tag{3.7}$$

$$Y_i, \; X_{ij} \in \{0, 1\} \; \forall \; i \in I \text{ and } j \in J \tag{3.8}$$

$$0 \leq S_i \leq s_{\max} \text{ and } \textit{integer} \; \forall \; i \in I. \tag{3.9}$$

In SBM, objective function (3.1) is the total annual costs of operating facil-

ities, transporting parts to customers and stocking parts at facilities. Constraints (3.2) and (3.3) are the UFL constraints, stating all demands must be allocated and demand can only be assigned to an open facility. Constraint (3.4) is the service level constraint which states that the total demand satisfied within the time-window aggregated across all customers and facilities must be at least α fraction of the total demand.

Since the details of the development of this constraint are in Candas and Kutanoglu (2006a), here we mention the underlying logic briefly: Due to the assumption that the individual demands at a facility, say i, are met in a FCFS fashion, they are satisfied proportional to the facility fill rate, β_i. The amount of demand satisfied from facility i "within the time-window" is $\beta_i \sum_{j \in J} \delta_{ij} d_j X_{ij}$. When this is aggregated across all facilities, we obtain the total demand satisfied within time-window, the left hand side of the constraint. (Constraint (3.4) is similar in spirit to the ones in Muckstadt (2005), who combines multiple facility (variable) fill rates to obtain an aggregate service level constraint).

Constraints (3.5) calculate λ_i for all i, the means of the lead time demands, which are used to compute the fill rates in (3.6). Due to the Poisson assumption of individual customer demands, the demand during lead time experienced at a facility is also Poisson with the mean that is equal to the aggregated mean demand assigned to the facility. Constraints (3.6) define the fill rates β_i for all i, as a function of mean lead times λ_i and stock levels S_i. The fill rate β_i at facility i is computed using the lost sales formula (see Zipkin (2000) for more details, and a comparison between the backorder fill rate and lost sales fill rate).

Constraints (3.7) state that every facility that has some demand allocated must have a positive stock level. Finally, constraints (3.8) state that variables Y_i and

X_{ij} are 0 or 1, and finally constraints (3.9) represent the integrality requirements of S_i, using $s_{\max}$ as the upper limit.

3.3 Solution Technique

SBM is a discrete, nonlinear, and nonconvex model, hence we cannot solve such models efficiently by direct optimization. In this section, we propose a substitution scheme that simplifies the model and uncovers a structure that makes the model amenable to be solved using direct optimization techniques and that leads to ideas for bounding the optimal solution value tightly. In the further development this substitution scheme will provide an equivalent convex nonlinear mixed integer problem which can be solved using an outer-approximation scheme.

3.3.1 Variable Substitution

We introduce decision variables θ_{ij}, and let $\theta_{ij} = \tau d_j \beta_i X_{ij}$, for all i and j. Also, we define $\Theta_i = \sum_{j \in J} \theta_{ij}$ for all i. Then,

$$\Theta_i = \sum_{j \in J} \theta_{ij} = \tau \beta_i \sum_{j \in J} d_j X_{ij} = \beta_i \lambda_i = \frac{\sum_{n=0}^{S_i - 1} \lambda_i^{n+1}/n!}{\sum_{n=0}^{S_i} \lambda_i^n/n!} \; \forall \; i \in I. \tag{3.10}$$

This shows that the value of Θ_i is the long term fraction of mean lead demand at facility i satisfied directly from stock. Now, we can replace (3.4) and (3.6) in SBM with the following constraints:

$$\sum_{i \in I} \sum_{j \in J} \delta_{ij} \theta_{ij} \geq \tau \alpha \sum_{j \in J} d_j \tag{3.11}$$

$$\Theta_i = \frac{\sum_{n=0}^{S_i - 1} \lambda_i^{n+1}/n!}{\sum_{n=0}^{S_i} \lambda_i^n/n!} \; \forall \; i \in I \tag{3.12}$$

$$\Theta_i = \sum_{j \in J} \theta_{ij} \ \forall \ i \in I \tag{3.13}$$

Note that (3.11) is equivalent to service level constraints, (3.4), under the substitution scheme. To handle the nonlinear substitution equation $\theta_{ij} = \tau d_j \beta_i X_{ij}$, we utilize the fact that X_{ij}'s are binary, and that $\tau \cdot d_j$ is the maximum possible value of θ_{ij}, and introduce the following equivalent linear constraints:

$$\theta_{ij} \geq \tau d_j(\beta_i + X_{ij} - 1) \ \forall \ i \in I \text{ and } j \in J \tag{3.14}$$

$$\theta_{ij} \leq \tau d_j(\beta_i + 1 - X_{ij}) \ \forall \ i \in I \text{ and } j \in J \tag{3.15}$$

$$\theta_{ij} \leq \tau d_j X_{ij} \ \forall \ i \in I \text{ and } j \in J. \tag{3.16}$$

3.3.2 Convexification

In this section, we investigate the generic version (suppressing subscript i) of the right hand sides of constraints (3.12) to handle their nonconvex nature. Consider:

$$g(\lambda, l) = \frac{\sum_{n=0}^{l-1} \lambda^{n+1}/n!}{\sum_{n=0}^{l} \lambda^n/n!}. \tag{3.17}$$

We first show that $g(\lambda, l)$ is a concave function of λ for a given stock level l. Figure 3.1 depicts g as a function of λ for a sample of stock levels.

Proposition 1 *For a given l, $g(\lambda, l)$ is an increasing and concave function of λ.*

Proofs of all propositions if required are given in the Appendix.

Let RBM be the relaxed version of SBM in which we replace (3.12) by the following constraints:

$$\Theta_i \leq \frac{\sum_{n=0}^{S_i-1} \lambda_i^{n+1}/n!}{\sum_{n=0}^{S_i} \lambda_i^n/n!} \ \forall \ i \in I. \tag{3.18}$$

Proposition 2 *The feasible region of RBM is convex in continuous variables.*

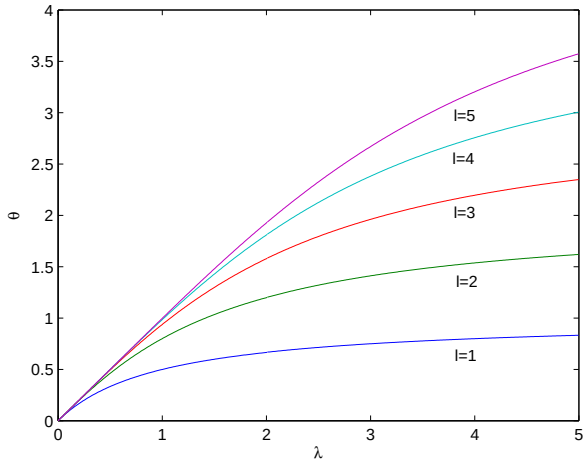


Figure 3.1: Θ as a function of λ for different stock levels, shown to be concave.

The proof of Proposition 2 follows directly from Proposition 1.

Proposition 3 *There exists an optimal solution of RBM which solves SBM. Furthermore, given an optimal solution of RBM we can construct an optimal solution to SBM in linear time.*

3.3.3 Outer Approximation

Before we introduce the outer approximation scheme for functions g, we generalize constraints (3.18) for variable stock levels. Let L be the set of stock levels that are considered in the model, i.e., $L = \{1, 2, \ldots, s_{\max}\}$, and let l be the index for this set. We now define a new set of binary decision variables, V_{il} for all i and l, each of which takes a value of 1 if facility i uses stock level l, and 0 otherwise. To model the variable nature of the stock levels, we replace constraints (3.18) with the following:

$$\Theta_i \leq \frac{\sum_{n=0}^{l} \lambda_i^{n+1}/n!}{\sum_{n=0}^{l} \lambda_i^{n}/n!} + M_l(1 - V_{il}) \; \forall \; l \in L, \text{ and } i \in I \tag{3.19}$$

$$S_i = \sum_{l \in L} l V_{il} \; \forall \; i \in I \tag{3.20}$$

$$\sum_{l \in L} V_{il} = 1 \; \forall \; i \in I. \tag{3.21}$$

Here, M_l is a large number for stock level l. When a specific stock level, say l' is chosen for a facility, say i, the second term on the right hand side of (3.19) disappears, and the corresponding constraint becomes active for $l = l'$ and redundant for $l \neq l'$ $\forall \; l \in L$. We can provide tight M_l values using the following result:

Proposition 4 *For each stock level $l \in L$, $\lim_{\lambda \to \infty} g(\lambda, l) = l$.*

Hence, we have $\Theta_i \leq g(\lambda_i, s_{\max}) \leq s_{\max}$. Then, a tight right-hand-side (independent of stock level l) for constraints (3.19) is $s_{\max}$. The idea here is to make the right hand side to add up to $s_{\max}$ when a constraint is inactive. Since we also know that $\Theta_i \leq g(\lambda_i, l) \leq l$ when $S_i = l$, we can replace M_l in constraints (3.18) with the following:

$$M_l = (s_{\max} - l) \; \forall \; l \in L. \tag{3.22}$$

Next we develop an approximation scheme using tangent lines to the non-linear term in the right-hand-side of the constraints in (3.19). Here, we once more exploit the fact that these functions are concave in $\lambda_i \; \forall \; i$. Under this scheme, we enclose g-curve from outside with its tangent lines at various points (see Figure 3.2 for an example). Consider $g(\lambda, l) = \frac{\sum_{n=0}^{l} \lambda^{n+1}/n!}{\sum_{n=0}^{l} \lambda^{n}/n!}$ as a function of λ for given stock level l at a facility that has mean lead time demand λ. We select a set K (indexed

by k) of finitely many points (each denoted by μ_k) on the λ-axis i.e., μ_k is the kth point in the interval $[0, \lambda^{\max}]$, where $\lambda^{\max}$ is an upper limit on the mean lead time demand that can be assigned to the facility. Let the lines tangent to the g-curve at $(\mu_k, g(\mu_k, l))$ be of the form $m_{kl}\lambda + b_{kl}$, with slopes m_{kl} and intercepts b_{kl}, which are computed as follows:

$$m_{kl} = g'(\mu_k, l),\ \forall\ k \in K,\ \text{and}\ l \in L \tag{3.23}$$

$$b_{kl} = g(\mu_k, l) - m_{kl}\mu_k,\ \forall\ k \in K,\ \text{and}\ l \in L \tag{3.24}$$

where $g'(\mu_k, l)$ is the first-order derivative of g with respect to λ evaluated at the corresponding points μ_k. For simplicity, we keep the set K and $\lambda^{\max}$ (and hence μ_k for all k) the same for all facilities. Set K and $\lambda^{\max}$ can be determined *a priori*. We can now write outer-approximation forms of constraints (3.19) as follows:

$$\Theta_i \leq m_{kl}\lambda_i + b_{kl} + (s_{\max} - l)(1 - V_{il})\ \forall\ i \in I,\ l \in L,\ \text{and}\ k \in K. \tag{3.25}$$

3.3.4 Valid Inequalities

We now introduce another variable substitution scheme that not only reduces the number of variables but also leads to important valid inequalities. The idea here is to use the binary representations of stock levels (S_i variables). Given a fixed value of $s_{\max}$, we need a total of $\lceil \log_2(s_{\max}) \rceil \times |I|$ binary variables to represent all possible values of stock levels at all facilities. We define $r = \lceil \log_2(s_{\max}) \rceil$ as the number of binary digits needed to represent $s_{\max}$. Given r, stock level l can be represented as

$$l = \sum_{n=1}^{r} 2^{n-1} e_{ln},\ \forall\ l = 1, 2, \ldots, s_{\max} \tag{3.26}$$

where e_{ln}'s are binary parameters representing the digits of the binary representation of l. For example, if $s_{\max} = 5$ ($r = 3$), and $l = 4$, then $e_{41} = 0$, $e_{42} = 0$, and $e_{43} = 1$.

We introduce a new set of variables U_{in}, $\forall\ i$ and $n = 1, 2, \ldots r$, that select the digits of the binary representation of stock level at facility i. Introduction of U_{in} completely eliminates the need for variables V_{il}. Then, we replace (3.19)-(3.21) with the following:

$$\begin{aligned} \Theta_i &\leq m_{kl}\lambda_i + b_{kl} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{in} \\ &\quad \forall\ i \in I,\ l \in L,\ \text{and } k \in K \qquad (3.27) \\ S_i &= \sum_{n=1}^{r} 2^{n-1}U_{in}\ \forall\ i \in I. \qquad (3.28) \end{aligned}$$

Using U_{in}, we introduce the following two sets of valid inequalities:

$$\begin{aligned} Y_i &\leq \sum_{n=1}^{r} U_{in}\ \forall\ i \in I \qquad (3.29) \\ Y_i &\geq U_{in}\ \forall\ i \in I \text{ and } n = 1, 2, \ldots, r. \qquad (3.30) \end{aligned}$$

The first set of valid inequalities (3.29) is logical relation stating that if Y_i is 1 then at least one of the U_{in} variables for facility i must be 1. These constraints can also be obtained by lifting (3.7) and making use of (3.3) (see Appendix 6.2.7). The second set of valid inequalities (3.30) is based on a logical reasoning to state that positive value of U_{in} for any n will cause Y_i to be positive. We can also add special valid inequalities for cases where $s_{\max} < \sum_{n=1}^{r} 2^n$. For example, when $s_{\max} = 5$, $U_{i3} + U_{i2} \leq 1$ is valid for all i.

3.3.5 Overall Mixed Integer Program

With all the developments in previous subsections, we write the overall model which is now a self-explanatory Mixed Integer Program (MIP):

$$\text{MIP: Minimize} \quad \sum_{i\in I} f_i Y_i + \sum_{i\in I}\sum_{j\in J} t_{ij} d_j X_{ij} + \sum_{i\in I} h_i S_i \tag{3.31}$$

$$\text{s.t.} \quad \sum_{i\in I} X_{ij} = 1 \ \forall\ j \in J \tag{3.32}$$

$$X_{ij} \leq Y_i \ \forall\ i \in I \text{ and } j \in J \tag{3.33}$$

$$\sum_{i\in I}\sum_{j\in J} \delta_{ij}\theta_{ij} \geq \tau\alpha \sum_{j\in J} d_j \tag{3.34}$$

$$\lambda_i = \tau \sum_{j\in J} d_j X_{ij} \ \forall\ i \in I \tag{3.35}$$

$$\Theta_i \leq m_{kl}\lambda_i + b_{kl} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{in} \quad \forall\ i \in I,\ l \in L, \text{ and } k \in K \tag{3.36}$$

$$\theta_{ij} \geq \tau d_j(\beta_i + X_{ij} - 1) \ \forall\ i \in I \text{ and } j \in J \tag{3.37}$$

$$\theta_{ij} \leq \tau d_j(\beta_i + 1 - X_{ij}) \ \forall\ i \in I \text{ and } j \in J \tag{3.38}$$

$$\theta_{ij} \leq \tau d_j X_{ij} \ \forall\ i \in I \text{ and } j \in J \tag{3.39}$$

$$\Theta_i = \sum_{j\in J} \theta_{ij} \ \forall\ i \in I \tag{3.40}$$

$$S_i = \sum_{n=1}^{r} 2^{n-1} U_{in} \ \forall\ i \in I \tag{3.41}$$

$$Y_i \leq \sum_{n=0}^{r} U_{in} \ \forall\ i \in I \tag{3.42}$$

$$Y_i \geq U_{in} \ \forall\ i \in I \text{ and } n = 1, 2, 3, \ldots, r \tag{3.43}$$

$$X_{ij}, Y_i, U_{in} \in \{0, 1\} \ \forall\ i \in I,\ j \in J \text{ and } n = 1, 2, 3, \ldots, r \tag{3.44}$$

$$0 \leq \beta_i \leq 1 \ \forall\ i \in I \tag{3.45}$$

$$\lambda_i, \Theta_i, \theta_{ij} \geq 0 \ \forall\ i \in I \text{ and } j \in J. \tag{3.46}$$

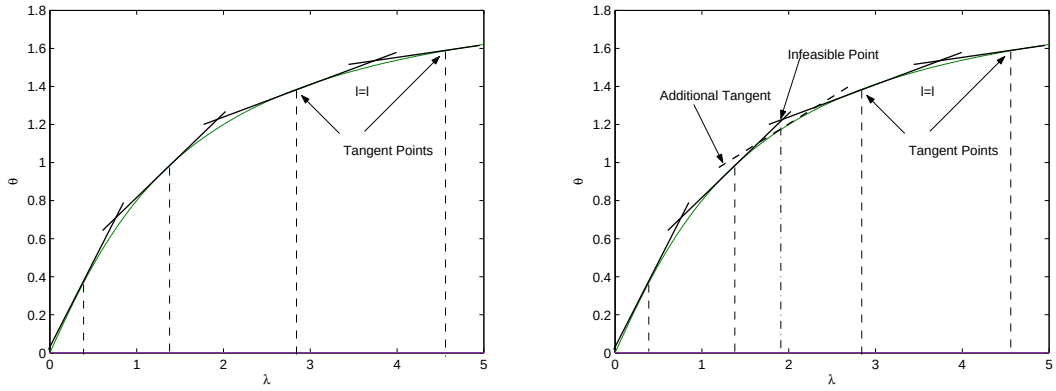


Figure 3.2: Outer approximation using tangent lines and cutting off infeasibility by an additional tangent line.

The idea here is to solve the problem with a set of initial tangential constraints in (3.36), and then expand the MIP formulation with new tangential constraints of the same form (3.36) as infeasibilities are detected with the current outer approximation. Below is complete listing of the outer-approximation algorithm.

Outer-Approximation Algorithm

1. Input: $t_{ij}, h_i, d_j, \delta_{ij}, m_{kl}, b_{kl},\ \forall\ i \in I,\ j \in J,\ k \in K,$ and, $l \in L$

2. Solve MIP (3.31)-(3.46)

3. Test the optimal solution's feasibility in SBM:

   (a) Let $\hat{\theta}_{ij},\ \hat{\lambda}_i,\ \hat{S}_i,\ \hat{X}_{ij}$ and $\hat{Y}_i$ be the optimal values of respective model variables.

   (b) Compute $\hat{\Theta}_i = \frac{\sum_{n=0}^{\hat{S}_i - 1} \hat{\lambda}_i^{n+1}/n!}{\sum_{n=0}^{\hat{S}_i} \hat{\lambda}_i^{n}/n!}$ for all $i \in I$

   (c) If $\sum_{j \in J} \hat{\theta}_{ij} \leq \hat{\Theta}_i$, for all $i \in I$, the current solution of MIP is optimal; go to Step 5.

4. Construct new tangential cuts:

(a) Set $k = |K| + 1$.

(b) For each i with $\sum_{j \in J} \hat{\theta}_{ij} > \hat{\Theta}_i$, do

i. Set $\mu_k = \hat{\lambda}_i$, $m_{kl} = g'(\lambda_i = \mu_k, l), \forall\ l \in L$, and $b_{kl} = g(\lambda_i = \mu_k, l) - m_{kl}\mu_k,\ \forall\ l \in L$.

ii. Add a set of constraints in the form (3.36) with new m_{kl} and b_{kl}'s for all i to MIP.

iii. Set $k = k + 1$ (Go to Step 4b).

(c) Append K to include the indices of the new tangents

(d) Go to Step 2

5. Output: $\hat{X}_{ij}, \hat{Y}_i, \hat{S}_i$ and $\hat{\beta}_i$ for all $i \in I$ and $j \in J$.

To detect the infeasibilities, we solve MIP to optimality using initial set K of tangents, and calculate the value of $\sum_{j \in J} \theta_{ij}$ using the optimal values $\hat{\theta}_{ij}$. If for some facility i the value $\sum_{j \in J} \hat{\theta}_{ij}$ is greater than the value of function $g(\lambda, l)$ evaluated at the facility's mean lead time demand $\hat{\lambda}_i = \tau \sum_{j \in J} d_j X_{ij}$ using optimal assignments $\hat{X}_{ij}$ and optimal stock level $\hat{S}_i$, then we cut off the current optimal solution by augmenting K with μ_{k+1} set equal to the facility's optimal mean lead time demand, $\hat{\lambda}_i$, and solve MIP again. (See Figure 3.2 as an illustration of adding such a tangent line). Whenever the optimal solution to the expanded MIP model is feasible to the original constraints, i.e., $\sum_{j \in J} \theta_{ij} \leq g(\hat{\lambda}_i, \hat{S}_i)$ for all $i \in I$, the solution is optimal to the original nonlinear model, signaling the stopping condition of the outer-approximation procedure.

3.4 Bounding Schemes

The variable substitution and linearization through outer-approximation along with the valid inequalities introduced in the previous sections make the original model amenable to direct optimization techniques as it is a mixed integer programming model. However, it is still a challenging problem to solve to optimality with direct optimization techniques, mainly due to its complexity and size. In this section, we develop a tight lower bounding scheme and also an upper bounding scheme that produces near-optimal heuristic solutions. Both schemes make use of the outer-approximation mechanism introduced earlier.

3.4.1 Lower Bounding: Dependency Relaxation

We use a special relaxation scheme to construct lower bounds. Under this scheme, we relax the dependency of fill rates on the customer demands that do not contribute towards the target service level. Specifically, we view the mean lead time demand λ_i at each i consisting of two components: (1) a component due to demands from customers that are within the time-window of the facility, and (2) another component due to demands from customers that are outside the time-window. We state $\lambda_i = \lambda_i^{\text{in}} + \lambda_i^{\text{out}}$ where

$$\lambda_i^{\text{in}} = \tau \sum_{j \in J} \delta_{ij} X_{ij} d_j \ \forall \ i \in I \tag{3.47}$$

$$\lambda_i^{\text{out}} = \tau \sum_{j \in J} (1 - \delta_{ij}) X_{ij} d_j \ \forall \ i \in I. \tag{3.48}$$

In the following, we show that all the models we introduced earlier (SBM, RBM, and MIP) can be modified to consider only λ_i^{in} to provide lower bounds on the original formulation.

Proposition 5 *The modified version of MIP in which constraints (3.35) is replaced by* $\{\lambda_i = \tau \sum_{j \in J} \delta_{ij} X_{ij} d_j \ \forall \ i \in I\}$*, is a relaxation of MIP (and of SBM).*

The main computational advantage of the relaxed model is that the use of $\lambda_i^{\text{in}} \ \forall \ i$ simplifies the modeling since the θ_{ij} variables are not needed any more; having Θ_i's is enough, as λ_i^{out} values do not affect the fill rates. This leads to a simplification of the relaxed model in which we do not have constraints (3.37)-(3.39). Under this relaxation scheme, the lower bound problem denoted by LBP is as follows:

$$\text{LBP:} \quad \text{Minimize} \quad \sum_{i \in I} f_i Y_i + \sum_{i \in I} \sum_{j \in J} t_{ij} d_j X_{ij} + \sum_{i \in I} h_i S_i \tag{3.49}$$

$$\text{s.t.} \quad \sum_{i \in I} X_{ij} = 1 \ \forall \ j \in J \tag{3.50}$$

$$X_{ij} \leq Y_i \ \forall \ i \in I \text{ and } j \in J \tag{3.51}$$

$$\sum_{i \in I} \Theta_i \geq \tau \alpha \sum_{j \in J} d_j \tag{3.52}$$

$$\lambda_i = \tau \sum_{j \in J} \delta_{ij} d_j X_{ij} \ \forall \ i \in I \tag{3.53}$$

$$\Theta_i \leq m_{kl} \lambda_i + b_{kl} + (s_{\max} - l) \sum_{n=1}^{r} (1 - e_{ln}) U_{in} \quad \forall \ l \in L, \ i \in I, \text{ and } k \in K \tag{3.54}$$

$$S_i = \sum_{n=1}^{r} 2^{n-1} U_{in} \ \forall \ i \in I \tag{3.55}$$

$$Y_i \leq \sum_{n=0}^{r} U_{in} \ \forall \ i \in I \tag{3.56}$$

$$Y_i \geq U_{in} \ \forall \ i \in I \text{ and } n = 1, 2, 3, \ldots, r \tag{3.57}$$

$$U_{in}, X_{ij}, Y_i \in \{0, 1\} \ \forall \ i \in I, \ j \in J \text{ and } n = 1, 2, 3, \ldots, r \tag{3.58}$$

$$\Theta_i \geq 0 \ \forall \ i \in I. \tag{3.59}$$

We call this model "dependency relaxation" since dependencies of β_i's on λ_i^{out} values have been relaxed. We still have to complete the outer-approximation process

for LBP to obtain the tightest possible lower bound. The solutions obtained from LBP will most likely be infeasible for the original model, not meeting the service level requirements when the true fill rates and the true mean lead time demands are considered, but will be useful for assessing the quality of upper bounds. Also, in the next section, we make use of the dependency relaxation to develop an upper bounding scheme.

3.4.2 Upper Bounding: α-boosting Scheme

The overall idea here is to solve a series of the relaxed models, each with temporarily increased (boosted) value of α, until the optimal solution to the "relaxed model with boosted α" satisfies the original service level constraint when tested with the true values of α, the mean lead time demands, and the fill rates. Note that the optimal solution of a relaxed model with boosted α is not a lower bound on the optimal value of the original model, but an upper bound whenever it is feasible – however it is expected to be feasible because artificial boosting of α may compensate the effects of dependency relaxation. This process effectively makes the lower bound model LBP a parametric model that we denote as $LBP(\alpha)$, where α is now the parameter. We denote the original target service level by α^O that we ultimately seek to satisfy in our final solution.

Let α^A be the actual service level achieved using the solution of $LBP(\alpha^O)$ with the true values of mean lead time demands and fill rates. If $\alpha^A \geq \alpha^O$, then the LBP solution is feasible (and hence optimal) to the original model. In this case, there is no need to boost α. Otherwise, if $\alpha^A < \alpha^O$, then we need to find the value of α greater than α^O, which when used in $LBP(\alpha)$ will produce a solution feasible to the original model. The goal is to find the tightest value for α so that the cost

is under control. For this, we try to find two values for α, one for which the LBP solution is feasible for the original problem, call α^U, and the other, call α^L, for which the the $LBP(\alpha)$ solution is infeasible. We then perform a bisection search between α^L and α^U to find the smallest α value (within ϵ tolerance) that yields a feasible solution to the original problem. To find the "tight" values for α^L and α^U we first keep boosting α in a controlled manner (using a scalar denoted by γ) until we obtain a feasible solution, recording the α values producing infeasible solutions as α^L and then perform a bisection search between α^L and α^U. We also define α^M which is the maximum service level a given network can achieve. It is important to know α^M since α cannot be boosted beyond it.

Below is the listing of upper bounding algorithm:

Upper Bounding Algorithm

1. $\alpha^M := 1 - \sum_{j \in J'} d_j \Big/ \sum_{j \in J} d_j$, where $J' = \{j \in J : \delta_{ij} = 0 \ \forall \ i \in I\}$
2. Search for α^L and α^U.
   (a) Set $u = 0$, pick the value of $\gamma \in (0, 1)$, and set $\alpha^L := \alpha^O$, $\alpha^U := \alpha^M$, and $\alpha := \alpha^O$.
   (b) Solve problem $LBP(\alpha)$, and set $u := u + 1$.
   (c) Calculate α^A for the solution of $LBP(\alpha)$.
   (d) If $\alpha^A \geq \alpha^O$, then set $\alpha^U := \alpha$ and go to Step 3. Otherwise,
      i. Set $\alpha^L := \alpha$
      ii. If $\alpha < \alpha^M$ then set $\alpha := \min\{\alpha^M, \alpha + \gamma(\alpha^O - \alpha^A)\}$, otherwise go to step 5
      iii. Go to Step 2b.
3. If $u = 1$, stop, and report the solution to $LBP(\alpha)$ as the optimal solution.

4. Perform bisection search between α^L and α^U

   (a) If $\alpha^U - \alpha^L < \epsilon$, then stop and report the LBP solution associated with α^U. Otherwise, set $\alpha := 0.5(\alpha^L + \alpha^U)$.

   (b) Solve $LBP(\alpha)$.

   (c) Calculate α^A for the solution of $LBP(\alpha)$.

   (d) If $\alpha^A \geq \alpha^O$, then set $\alpha^U := \alpha$. Otherwise, set $\alpha^L := \alpha$. Go to Step 4a.

5. Exit and report no feasible solution found.

Note that step 5 is provided to take care of instances for which α-boosting algorithm may not return a feasible solution. Such instances can be generated by requiring service level α^O very close to α^M, the maximum service level possible with very high cost, by opening all locations and stocking them with very high stock levels. The algorithm may not obtain feasible solutions for such instances since the values of α between α^O and α^M may not be sufficient for boosting required to compensate dependency relaxation. To obtain feasible solutions for such instances, direct optimization applied to the original MIP model described in Section 3.3.5 must be used. However, such a need does not arise for any of the instances we solve, which signals the relative robustness of the overall approach.

3.5 Computational Study

We now discuss the computational study conducted to evaluate efficacy of the lower bounding and upper bounding schemes. The study uses two sets of data; the first set consists of small-size instances that are generated randomly, and the second set consists of larger instances that are based on real data from a service parts division

of a computer hardware manufacturer. We first explain how the problems are generated. We then report our results obtained by solving both the lower bounding and upper bounding optimization problems using CPLEX 9.0. CPLEX is also used to directly solve the MIP with outer-approximation. With a set of preliminary experiments, we fine tuned several options in CPLEX to obtain favorable computation times across all experiments. The same options are used for both the original MIP problem and $LBP(\alpha)$ used in the lower bounding and upper bounding schemes. In our preliminary experimentation, we found that the following CPLEX options helped improve the solution times considerably (over default options):

$probe$ 1	(Limited probing)
$precompress$ 1	(Precompressor ON)
$mircuts$ -1	(Do not generate MIR cuts)
$nodesel$ 2	(Best-estimate search)
$varsel$ 2	(Branch based on pseudo costs)
$lpmethod$ 4	(LP Method - Barrier)
$fraccuts$ 2	(Generate Gomory fractional cuts aggressively)
$mipemphasis$ 2	(Emphasize optimality over feasibility).

We use a PC with dual Xeon 1.8 GHz processors with 1 GB RAM running Suse Linux operating system for all our computations.

3.5.1 Test Problem Data

The small instances are generated using uniform random numbers. Each small problem instance has 15 facilities and 50 demand points, denoted by 15×50 for

short later. Customer and facility locations are generated randomly on a grid of 150×150. Fixed cost of opening a facility f_i are set to 0 and transportation costs are proportional to distance, setting t_{ij} at one tenth of the Euclidean distance between facility i and customer j, and rounding it to a positive integer. Mean demand values (d_j) are generated uniformly within (1,3). Assuming the mean demand values are for a year, we assume 7 days of replenishment lead times for all facilities. Time-window indicators δ_{ij} are determined using a circle around each facility location with a radius of 40 units, i.e., set $\delta_{ij} = 1$, if the Euclidean distance between i and j is less than 40, 0 otherwise. The maximum allowed stock level s_{max} is 5, which is more than enough to obtain fill rates very close to 100% at all facilities. We initialize the outer-approximation of the graphs of Θ_i with 13 tangent lines at the breakpoints of the mean lead time demand that correspond to the percentages of the maximum possible mean lead time demand for a facility. The maximum mean lead time demand is $\Lambda = \tau \sum_{j \in J} d_j$ and the μ_k's are found as follows:

$$\mu_k = p_k \cdot \Lambda, \ \forall \ p_k \in \{.05, 0.10, 0.15, 0.20, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$$

We experiment with 5 levels of holding cost h, namely 1, 10, 20, 50, or 100. We use three target service levels (40%, 60%, or 80%) of the maximum possible service (α^M) . Replicating each combination of the holding cost and target service level 3 times with randomly generated demands, demand point locations, and facility locations, we have a total of 45 small instances. The maximum possible service level is 1 as each customer has at least one facility covering its demand in every instance (i.e., $\alpha^M = 1$ for all instances). A time limit of 900 seconds is used for CPLEX to stop its computation. If the optimality is not proved by the time limit, then the best integer feasible solution is reported.

The larger instances are based on real data provided by a service parts logistics group at a large computer hardware manufacturer. We use 6 representative networks (demand points and candidate facility locations), representing different regions in the United States. As the sizes of the networks are different depending on the region, we show the numbers of facility locations and demand points in the tables listing the results. We use the actual transportation costs and δ_{ij} values provided in the real data. The data have three service time-windows (representing 2 hr, 4 hr, and 12 hr response times), which are used here separately for each instance, to analyze the effects of the time-window on our results. As the time-window gets longer, representing lengthening the radius around each stocking facility, more δ_{ij} values become 1 and it becomes "easier" to satisfy the service level, as there are more candidate stocking facilities that can provide service to each demand point within the time-window. As the original data have demands for individual parts that are extremely low (which do not really challenge the model to stock more than one unit at a facility), we generate somewhat inflated (but still low) mean customer demands (d_j's) uniformly within (1,3). We experiment with three levels of holding costs (50, 100, 200) and three levels of service, which are adjusted according to the maximum possible service level in each network, namely (40%, 60% and 80%) of the maximum possible service α^M. The time limit for these instances is set to be 1800 seconds. In all the experiments, we use $\gamma = 1$, $\epsilon = 0.0005$ for the upper bounding scheme, and again the fixed costs of facilities, f_i's, are set to 0.

3.5.2 Results: Small Problems

The first set of results shows the effectiveness of binary representation of stock levels (and their associated valid inequalities) discussed in Section 3.3.4. Table 3.1

shows the saving in computation times achieved by using the binary representation and associated valid inequalities in the dependency-relaxation based lower bounding problem, $LBP(\alpha = \alpha^O)$, solved for all the small problem instances, first solved with original stock level variables V_{il}'s and then with binary representation variables U_{in}'s. The table lists the instances with their associated "replication (or seed)" (a, b, and c), holding cost (h), and the target service level (α^O). Both models solve all the 45 instances to optimality and get the same objective function value, finishing the outer-approximation along the way with added tangent lines if necessary. Here, the main dimension of comparison is the computation time. The model with binary representation and valid inequalities solves all 45 instances within 10 minutes, the model without these improvements takes 31 minutes. Although the model with binary representation takes one third of the original model's computation time, the maximum relative saving in computation could be as high as 800% (instance 3b). With these observed computational savings, we use the model with binary representation and associated valid inequalities to obtain the rest of the results.

Computational results for the small instances are presented for two cases of fixed cost of opening a facility (f_i): (1) zero fixed cost ($f_i = 0 \ \forall \ i \in I$) (2) nonzero fixed cost ($f_i = 1000 \ \forall \ i \in I$). The first two tables 3.2 and 3.3 present the results for case 1 and the next two tables 3.4 and 3.5 for the case 2. Table 3.2 shows the best lower and upper bounds obtained using our bounding schemes and compares them with the outer-approximation based direct optimization. The first 3 columns in this table show the characteristics of problem instance. The next 4 columns list the results obtained by the bounding schemes. LB is the best lower bound obtained solving $LBP(\alpha)$, UB is the best upper bound obtained through α-boosting, G(UL)% is the percentage gap between UB and LB, computed as $100(\frac{UB-lB}{LB})$, and α^A is the

achieved time-based service level by the upper bound solution. The last 4 columns list the results obtained by direct optimization. OBJ is the objective function value obtained by CPLEX, which is either the optimal solution value or the best integer solution value if the time limit of 900 seconds is reached before proving optimality, α^A is the achieved service level by the direct optimization solution, G(OBJ)% is the percentage gap between OBJ and the CPLEX's best linear programming relaxation based lower bound, and G(OU)% is the percentage gap between OBJ and UB, computed as $100(\frac{OBJ-UB}{OBJ})$. Table 3.3 solution characteristics (number of facilities open and total stock) and CPU time. The first 3 columns in this table again show the characteristics of problem instance. The next 3 columns list the results obtained by the bounding schemes, $\sum Y_i$ is the number of stocked facilities out of 15 facilities, $\sum S_i$ is the total stock across all facilities, and TIME is the computation time for both lower and upper bounds in seconds. The next 3 columns show respective results for direct optimization. Rows in the tables are divided into three blocks (of 15 rows each) representing three different random seeds. These 45 instances cover variety of difficulty levels; in general, an instance with a higher service level and higher holding cost tends to be more difficult. Hence, in each 15-row block, the first problem usually represents an easier setting while the last problem is the hardest.

Analyzing the results of the α-boosting first, we observe that in general the lower bound provided by the proposed dependency relaxation is very good, leading to feasible (and provably optimal) solutions to the original problem in some instances (these are shown with 0% G(UL)). The results also show that the upper bounds are very close to the lower bounds for most instances producing an average gap of 1.42% between UB and LB. In general, the higher service levels and higher holding costs lead to wider gaps between LB and UB, up to 11% in the tested problems. However,

we believe that this is mainly due to the tightness of LB, not UB, as it will be clear with the comparison between UB and OBJ. As shown with the achieved service levels, sometimes the problem does not need an explicit service level constraint, as the economics of the problem (tradeoff between transportation and holding costs) leads to a solution that already achieves (overshoots) the target service level. That is why increasing the service level further leads to the same solution; unless increase in the target service level or in the holding cost leads to a change in the solution. As observed with the number of stocked facilities and the total stock levels, the instances across three seeds lead to stocking from 3 to 12 facilities (out of 15) with stock levels per facility from 1 to 3, which are the realistic ranges of values to solve a wide variety of nontrivial problems.

The maximum CPU time for the proposed method of α-boosting is 363 seconds, and the average CPU time over 45 instances is 47 seconds. In contrast, the average CPU time for direct optimization is 286 seconds, while the time limit of 900 seconds (15 minutes) is reached with direct optimization for almost every instance with high service levels and high holding costs.

Looking at the direct optimization results, we observe that majority of instances are solved to optimality within the time limit, but there are several with up to 30% final gap between the best integer solution and the lower bound. There are 3 instances (15a, 12b, and 15c) for which the outer-approximation process could not finish adding tangent lines to eliminate all infeasibilities due to the time limit, reporting integer but infeasible solutions (denoted by an * at achieved service levels). There are a handful of instances that direct optimization leads to better objective function values than the α-boosting method (negative values in the last column, G(OU)%); in four of these the improvement due to direct optimization is less than

1%, and in one instance (15c), it is 4.22%. Note that the values in this column also confirm that the upper bounds' actual optimality gaps are a lot better than the ones estimated with the dependency relaxation lower bounds, especially for the instances that are solved with direct optimization to optimality. (Compare G(UL)% and G(OU)% for which the CPU time for direct optimization is less than 900 seconds).

There are cases where the dependency relaxation lower bounds are very strong. For example, direct optimization reports final duality gap of 18.09% for instance 15b while the dependency relaxation lower bound yields 0.08% (note that the direct optimization OBJ and our UB are very close). However, we conjecture that the strength of the lower bound is very much data dependent, as the main relaxation is due to ignoring the contribution of the demands from customers that are outside the time-window on the mean lead time demands of the facilities. As it turns out, this portion could be significant to ignore for some randomly generated problems, leading to not so tight lower bounds. In summary, for these small instances the upper bounds obtained through α-boosting can be slightly worse than the solutions obtained from direct optimization for some instances, but the overall technique is significantly beneficial in terms of computation time.

Table 3.4 and 3.5 have the exact same structure described above for Table 3.2 and 3.3, they present results for the case 2. These results with nonzero fixed costs are not significantly different from those with zero fixed costs. The bounding schemes and direct optimization can be compared along two directions, (1) time and (2) solution quality. Direct optimization may give better solution with larger duality gaps taking more CPU times, however our bounding schemes provide tighter lower bounds and very competitive upper bounds much faster. The results in Table 3.4 and 3.5 are presented here to show that our bounding schemes are unaffected by

fixed costs of opening facilities. Rest of the computation based on real life larger instances assumes $f_i = 0,\ \forall\ i$.

3.5.3 Results: Large Problems

Computational results for the larger instances are organized in 6 tables, 2 tables for each time-window. The tables have identical column structure used to present results of small instances. The tables 3.6 - 3.11 compare performance of our bounding schemes with direct optimization using larger instances. This data set based on real data have three time-windows (TW) that define three sets of δ_{ij} values (denoted as TW1 for 2 hr, TW2 for 4 hr, and TW3 for 12 hr time-windows). We present results for 6 "networks" (denoted by a, b, c, d, e, and f in the tables) representing actual service regions with varying number of stocking locations and demand points. For each network, we generate 9 instances using 3 different service levels and 3 different holding costs. All tables have 6 blocks of 9 rows each. With respect to time, TW1 represents the most restrictive setting (with few facilities for each demand point that can provide the service within the time-window), TW2 the moderately restrictive setting (with more facilities with the capability of serving each customer within the time-window), and TW3 the loose setting (every facility can provide the service to every customer within the time-window, provided it has the part available, i.e., $\delta_{ij} = 1$ for all i and j). As we widen the time-window, more customers can be served within the time-window (causing more δ_{ij}'s to be 1), making it easier to achieve a given service level. Hence, with a wider time-window, a given network will achieve a larger α^M. To have relatively the same level of difficulty of satisfying the target service level across multiple time-windows and to avoid infeasibilities, we adjust the ultimate target service level (α^O) according to what is achievable (feasible) in the

network (α^M) using the same scale ψ parameter in all time-window settings, and making α^O time-window dependent.

Tables 3.6 and 3.7 present results for the most strict time-window (TW1). Due to the structure of these problems, the dependency relaxation solutions become feasible (and optimal) for the original problem, producing 0% gap between lower and upper bounds in majority of the instances. This is especially true for low service levels and low holding costs. The overall gap is extremely small (0.24%), with a maximum of 1.71%. Direct optimization gives an average final gap of 0.11% with a maximum of 1.73%, producing very close results to the α-boosting method. However, as reported under CPU times, direct optimization may take up to half an hour to obtain some of these results (see instances with high holding costs) with an overall average time of about 8 minutes per instance, whereas the α-boosting method takes 16 seconds per problem on average, a little over 5 minutes at maximum. In terms of solution quality comparison, direct optimization has slightly better objective function values for a few of instances (i.e., listed with negative gaps in the last column; total of 8 such instances; for 4 of these direct optimization reports an infeasible solution). In all other instances, the α-boosting method and direct optimization give the exact same solution, yielding 0% gap in the last column. Average of this column is -0.09%, signaling a minor benefit for direct optimization, but we know that 4 out of 54 direct optimization solutions are infeasible.

Although TW1 is the most restrictive time-window, instances in Table 3.6 are not the hardest instances to solve. Instances in Tables 3.8 and 3.9 for TW2, which have more δ_{ij}'s 1 compared to TW1 representing moderately restricted instances, turn out to be the hardest instances among all the problems tested. For this time-window, dependency relaxation solutions are usually infeasible for all net-

works, except network c, requiring the α boosting stage of the overall scheme. The average gap between lower bounds and upper bounds is 0.51%, with a maximum of 3.16%. Note that part of these gaps are due to the weakness of the lower bounds, not the quality of the upper bounds, observed by comparing the upper bounds with the direct optimization results. Direct optimization yields average final gap of 0.38%, with some without optimality proven (an integer feasible solution reported), but direct optimization is not able to produce a feasible solution for 19 of 54 instances within the given time limit. In most of these time consuming instances, the reported solutions are infeasible because the outer-approximation's tangent line addition is not completed by the time limit. Especially when the holding costs are moderate or high, the problems become extremely hard for direct optimization. Again, checking the last column's gap between the direct optimization objective and the α-boosting upper bound may be misleading, due to these infeasible direct optimization solutions. Comparing the solution times, we can conclude that the α-boosting method provides not only high quality feasible solutions to all instances but also takes only about 25% computation time of direct optimization on these set of hardest instances.

Tables 3.10 and 3.11 present the results of the large instances with the widest time-window, TW3, representing 12 hr response time. As the networks represent relatively small geographic regions, this leads to δ_{ij} being 1 for i and j pairs, i.e., every facility can serve each customer within the 12-hr time-window if assigned. These instances are the easiest, as the dependency relaxation is not a "relaxation" anymore, because there is no ignored demand that is coming from customers that are "outside the time-window," i.e., $\lambda_i^{\text{in}} = \lambda_i$ for all i. It turns out that these instances are also easy for direct optimization, hence both the α-boosting method (without really any boosting stage, as the relaxation solution solves the problem) and direct

optimization provide identical results, and every instance is solved to optimality by both methods. The difference in the CPU time taken by the two methods is due to the difference in the models they solve: The model in the dependency relaxation is $LBP(\alpha^O)$ with the compact formulation using Θ_i's, whereas the direct optimization model is the one with explicit θ_{ij}'s. The α-boosting method solves all 54 instances in 16 minutes while direct optimization takes about 32 minutes. Although instances with such time-windows that are large enough to have all δ_{ij}'s to be 1 are easier, they provide an idea for another relaxation for the general case. In this case, setting $\delta_{ij} = 1 \; \forall \; i$ and j will be another relaxation, which as we observe is easier to solve. We leave the further investigation of this "time-window relaxation," especially analyzing how it compares to the dependency relaxation as a future work item.

3.6 Summary

In this chapter we introduced an efficient model and solution schemes for the integrated problem for a single part type, for the special case of fill rate computation formula. The modeling and solution schemes are shown to be very effective. We take advantage of the special structure of the problem hidden in the concave Θ functions. Due to considering only a single part type for a single product we ignored part commonality which is an important aspect of the integrated problem. The modeling and solution schemes presented in the chapter can be exploited with part commonality (single part type common to multiple products) also. In the next chapter we discuss modeling issues that arise due to part commonality.

Table 3.1: Computational time savings achieved by using binary representation and associated valid inequalities – shown for small 15x50 problem instances with zero fixed cost ($f_i = 0 \; \forall \; i$)

Problem				w/o Valid Inequalities	w/ Valid Inequalities	Comparison
n	h	α^O	LB	CPU Time	CPU Time	% Time Saved
1a	1	0.4	232	0.11	0.10	10.00
2a	1	0.6	232	0.11	0.11	0.00
3a	1	0.8	232	0.11	0.11	0.00
4a	10	0.4	314	0.15	0.12	25.00
5a	10	0.6	314	0.17	0.11	54.55
6a	10	0.8	321	5.05	1.92	163.02
7a	20	0.4	369	0.15	0.12	25.00
8a	20	0.6	369	0.16	0.11	45.45
9a	20	0.8	410	22.74	6.89	230.04
10a	50	0.4	491	0.20	0.15	33.33
11a	50	0.6	491	0.20	0.15	33.33
12a	50	0.8	650	65.17	22.88	184.83
13a	100	0.4	659	4.54	4.02	12.94
14a	100	0.6	691	3.78	2.80	35.00
15a	100	0.8	1050	88.74	37.31	137.85
1b	1	0.4	279	0.12	0.11	9.09
2b	1	0.6	279	0.12	0.11	9.09
3b	1	0.8	280	1.72	0.19	805.26
4b	10	0.4	368	0.15	0.12	25.00
5b	10	0.6	368	0.17	0.12	41.67
6b	10	0.8	385	29.88	7.73	286.55
7b	20	0.4	443	0.19	0.14	35.71
8b	20	0.6	443	0.15	0.14	7.14
9b	20	0.8	489	16.77	8.32	101.56
10b	50	0.4	583	0.14	0.12	16.67
11b	50	0.6	626	10.00	9.22	8.46
12b	50	0.8	789	140.25	23.63	493.53
13b	100	0.4	740	1.00	5.76	-82.64
14b	100	0.6	926	35.34	31.44	12.40
15b	100	0.8	1289	83.37	44.50	87.35
1c	1	0.4	290	0.11	0.11	0.00
2c	1	0.6	290	0.11	0.11	0.00
3c	1	0.8	296	11.10	8.19	35.53
4c	10	0.4	366	0.14	0.10	40.00
5c	10	0.6	366	0.18	0.11	63.64
6c	10	0.8	427	90.53	27.84	225.18
7c	20	0.4	416	0.14	0.12	16.67
8c	20	0.6	426	3.13	1.62	93.21
9c	20	0.8	556	545.40	78.22	597.26
10c	50	0.4	524	0.14	0.12	16.67
11c	50	0.6	588	7.48	5.47	36.75
12c	50	0.8	916	340.24	122.57	177.59
13c	100	0.4	674	1.47	0.88	67.05
14c	100	0.6	838	9.43	10.08	-6.45
15c	100	0.8	1516	383.79	164.40	133.45

Table 3.2: Results of small (15×50) problem instances with $f_i = 0 \; \forall \; i$

Problem			Bounding schemes				Direct optimization			
n	h	α^O	LB	UB	G(UL)%	α^A	OBJ	α^A	G(CPX)%	G(OU)%
1a	1	0.4	232	232	0.00	0.806	232	0.816	0.00	0.00
2a	1	0.6	232	232	0.00	0.806	232	0.816	0.00	0.00
3a	1	0.8	232	232	0.00	0.806	232	0.821	0.00	0.00
4a	10	0.4	314	314	0.00	0.738	314	0.734	0.00	0.00
5a	10	0.6	314	314	0.00	0.735	314	0.739	0.00	0.00
6a	10	0.8	321	321	0.00	0.802	321	0.801	0.00	0.00
7a	20	0.4	369	369	0.00	0.654	369	0.654	0.00	0.00
8a	20	0.6	369	369	0.00	0.653	369	0.654	0.00	0.00
9a	20	0.8	410	411	0.23	0.801	411	0.802	0.00	0.00
10a	50	0.4	491	491	0.00	0.584	491	0.584	0.00	0.00
11a	50	0.6	491	519	5.32	0.654	519	0.653	0.00	0.00
12a	50	0.8	650	652	0.35	0.805	652	0.805	5.63	0.00
13a	100	0.4	659	665	0.80	0.414	665	0.414	0.00	0.00
14a	100	0.6	691	769	10.09	0.653	769	0.653	7.98	0.00
15a	100	0.8	1050	1052	0.22	0.803	1053	0.799*	13.49	0.01
1b	1	0.4	279	279	0.00	0.778	279	0.778	0.00	0.00
2b	1	0.6	279	279	0.00	0.776	279	0.779	0.00	0.00
3b	1	0.8	280	280	0.00	0.802	280	0.802	0.00	0.00
4b	10	0.4	368	368	0.00	0.737	368	0.734	0.00	0.00
5b	10	0.6	368	368	0.00	0.737	368	0.736	0.00	0.00
6b	10	0.8	385	388	0.81	0.806	388	0.802	0.00	0.00
7b	20	0.4	443	443	0.00	0.651	443	0.653	0.00	0.00
8b	20	0.6	443	443	0.00	0.666	443	0.653	0.00	0.00
9b	20	0.8	489	490	0.21	0.800	490	0.800	0.00	0.00
10b	50	0.4	583	583	0.00	0.430	583	0.430	0.00	0.00
11b	50	0.6	626	629	0.43	0.602	628	0.601	0.00	-0.15
12b	50	0.8	789	790	0.13	0.800	790	0.799*	6.93	0.00
13b	100	0.4	740	783	5.52	0.430	783	0.430	0.00	0.00
14b	100	0.6	926	929	0.29	0.600	928	0.601	4.13	-0.10
15b	100	0.8	1289	1290	0.08	0.800	1293	0.799*	18.09	0.23
1c	1	0.4	290	290	0.00	0.679	290	0.680	0.00	0.00
2c	1	0.6	290	290	0.00	0.683	290	0.679	0.00	0.00
3c	1	0.8	296	298	0.67	0.802	297	0.800	0.00	-0.34
4c	10	0.4	366	366	0.00	0.605	366	0.599	0.00	0.00
5c	10	0.6	366	366	0.00	0.605	366	0.604	0.00	0.00
6c	10	0.8	427	438	2.52	0.801	434	0.800	4.72	-0.85
7c	20	0.4	416	416	0.00	0.541	416	0.542	0.00	0.00
8c	20	0.6	426	426	0.00	0.605	426	0.606	0.00	0.00
9c	20	0.8	556	579	3.88	0.804	581	0.800	13.74	0.40
10c	50	0.4	524	533	1.62	0.405	533	0.408	0.00	0.00
11c	50	0.6	588	606	2.86	0.606	606	0.606	1.76	0.00
12c	50	0.8	916	999	8.25	0.804	1003	0.801	26.02	0.39
13c	100	0.4	674	683	1.26	0.405	683	0.406	0.00	0.00
14c	100	0.6	838	906	7.43	0.606	906	0.606	7.98	0.00
15c	100	0.8	1516	1699	10.74	0.803	1630	0.800	30.09	-4.22

* infeasible solutions

Table 3.3: Results of small (15×50) problem instances with $f_i = 0 \; \forall \; i$

Problem			Bounding schemes			Direct optimization		
n	h	α	$\sum Y_i$	$\sum S_i$	TIME	$\sum Y_i$	$\sum S_i$	TIME
1a	1	0.4	12	12	0	12	12	0
2a	1	0.6	12	12	0	12	12	0
3a	1	0.8	12	12	0	12	12	0
4a	10	0.4	7	7	0	7	7	0
5a	10	0.6	7	7	0	7	7	0
6a	10	0.8	9	9	12	9	9	53
7a	20	0.4	5	5	0	5	5	0
8a	20	0.6	5	5	0	5	5	0
9a	20	0.8	9	9	26	9	9	388
10a	50	0.4	4	4	0	4	4	0
11a	50	0.6	5	5	18	5	5	287
12a	50	0.8	7	8	82	7	8	900
13a	100	0.4	3	3	140	3	3	126
14a	100	0.6	5	5	58	5	5	900
15a	100	0.8	7	8	127	7	8	900
1b	1	0.4	15	15	0	15	15	0
2b	1	0.6	15	15	0	15	15	0
3b	1	0.8	15	16	0	15	16	2
4b	10	0.4	9	9	0	9	9	0
5b	10	0.6	9	9	0	9	9	0
6b	10	0.8	9	11	15	9	11	36
7b	20	0.4	7	7	0	7	7	0
8b	20	0.6	7	7	0	7	7	0
9b	20	0.8	8	10	30	8	10	252
10b	50	0.4	4	4	0	4	4	0
11b	50	0.6	6	6	45	6	6	272
12b	50	0.8	8	10	61	8	10	900
13b	100	0.4	4	4	47	4	4	384
14b	100	0.6	6	6	139	6	6	900
15b	100	0.8	8	10	122	8	10	900
1c	1	0.4	11	11	0	11	11	0
2c	1	0.6	11	11	0	11	11	0
3c	1	0.8	11	19	35	11	18	112
4c	10	0.4	6	6	0	6	6	1
5c	10	0.6	6	6	0	6	6	1
6c	10	0.8	7	14	159	8	14	900
7c	20	0.4	5	5	0	5	5	1
8c	20	0.6	6	6	1	6	6	12
9c	20	0.8	7	14	224	7	14	900
10c	50	0.4	3	3	1	3	3	61
11c	50	0.6	6	6	35	6	6	900
12c	50	0.8	7	14	269	7	14	900
13c	100	0.4	3	3	9	3	3	48
14c	100	0.6	6	6	78	6	6	900
15c	100	0.8	7	14	363	7	13	900

Table 3.4: Results of small (15×50) problem instances with $f_i = 1000 \; \forall \; i$

Problem			Bounding schemes				Direct optimization			
n	h	α^O	LB	UB	G(UL)%	α^A	OBJ	α^A	G(CPX)%	G(OU)%
1a	1	0.4	2443	2444	0.04	0.405	2444	0.404	0.00	0.00
2a	1	0.6	3365	3366	0.03	0.601	3366	0.601	0.00	0.00
3a	1	0.8	4299	4300	0.02	0.809	4300	0.809	0.00	0.00
4a	10	0.4	2479	2489	0.40	0.404	2489	0.404	0.00	0.00
5a	10	0.6	3419	3435	0.45	0.608	3429	0.601	0.00	-0.15
6a	10	0.8	4371	4381	0.23	0.803	4381	0.809	0.00	0.00
7a	20	0.4	2519	2539	0.79	0.405	2539	0.405	0.00	0.00
8a	20	0.6	3479	3505	0.73	0.606	3499	0.601	1.56	-0.15
9a	20	0.8	4451	4471	0.45	0.803	4471	0.809	0.00	0.00
10a	50	0.4	2639	2689	1.89	0.405	2689	0.405	0.78	0.00
11a	50	0.6	3659	3715	1.51	0.606	3701	0.599	4.65	-0.36
12a	50	0.8	4691	4741	1.07	0.809	4741	0.804	0.00	0.00
13a	100	0.4	2824	3024	7.08	0.417	2939	0.405	1.19	-2.81
14a	100	0.6	3959	4065	2.66	0.606	3996	0.600	3.83	-1.68
15a	100	0.8	5091	5191	1.96	0.809	5191	0.804	1.74	0.00
1b	1	0.4	2533	2535	0.08	0.409	2535	0.406	0.00	0.00
2b	1	0.6	3447	3449	0.06	0.605	3449	0.602	0.00	0.00
3b	1	0.8	6340	6342	0.03	0.805	6341	0.800	2.12	-0.02
4b	10	0.4	2560	2580	0.78	0.409	2580	0.437	0.00	0.00
5b	10	0.6	3510	3530	0.57	0.609	3525	0.600	0.00	-0.14
6b	10	0.8	6454	6464	0.15	0.800	6464	0.800	3.93	0.00
7b	20	0.4	2590	2630	1.54	0.409	2630	0.412	0.00	0.00
8b	20	0.6	3580	3620	1.12	0.605	3605	0.600	0.00	-0.41
9b	20	0.8	6574	6594	0.30	0.803	6594	0.800	2.99	0.00
10b	50	0.4	2680	2780	3.73	0.415	2780	0.410	2.05	0.00
11b	50	0.6	3790	3890	2.64	0.609	3845	0.600	0.00	-1.15
12b	50	0.8	6934	6954	0.28	0.800	6954	0.800	6.05	0.00
13b	100	0.4	2830	3030	7.07	0.403	3030	0.403	5.08	0.00
14b	100	0.6	4140	4340	4.83	0.609	4245	0.600	2.95	-2.18
15b	100	0.8	7534	7554	0.26	0.803	7654	0.802	9.91	1.32
1c	1	0.4	2460	2462	0.08	0.414	2462	0.415	0.00	0.00
2c	1	0.6	3383	3385	0.06	0.601	3385	0.602	0.00	0.00
3c	1	0.8	6325	6326	0.02	0.800	6326	0.800	0.00	0.00
4c	10	0.4	2496	2530	1.35	0.421	2516	0.415	0.00	-0.55
5c	10	0.6	3443	3453	0.29	0.602	3453	0.601	0.00	0.00
6c	10	0.8	6460	6470	0.15	0.800	6470	0.800	0.61	0.00
7c	20	0.4	2536	2580	1.73	0.421	2576	0.415	0.00	-0.15
8c	20	0.6	3503	3523	0.57	0.607	3523	0.602	0.37	0.00
9c	20	0.8	6610	6630	0.30	0.800	6630	0.800	1.42	0.00
10c	50	0.4	2656	2730	2.78	0.421	2729	0.404	1.29	-0.04
11c	50	0.6	3683	3733	1.36	0.601	3733	0.602	1.18	0.00
12c	50	0.8	7060	7110	0.71	0.800	7110	0.800	4.27	0.00
13c	100	0.4	2856	2980	4.33	0.421	2980	0.412	3.60	0.00
14c	100	0.6	3983	4083	2.51	0.604	4083	0.601	2.70	0.00
15c	100	0.8	7810	7910	1.29	0.801	7845	0.800	6.89	-0.83

* infeasible solutions

Table 3.5: Results of small (15×50) problem instances with $f_i = 1000 \ \forall \ i$

Problem			Bounding schemes			Direct optimization		
n	h	α	$\sum Y_i$	$\sum S_i$	TIME	$\sum Y_i$	$\sum S_i$	TIME
1a	1	0.4	2	5	35	2	5	81
2a	1	0.6	3	7	59	3	7	217
3a	1	0.8	4	9	18	4	9	399
4a	10	0.4	2	5	56	2	5	73
5a	10	0.6	3	7	85	3	7	599
6a	10	0.8	4	9	39	4	9	743
7a	20	0.4	2	5	70	2	5	132
8a	20	0.6	3	7	67	3	7	900
9a	20	0.8	4	9	54	4	9	752
10a	50	0.4	2	5	70	2	5	900
11a	50	0.6	3	7	55	3	6	900
12a	50	0.8	4	9	47	4	9	656
13a	100	0.4	2	5	86	2	5	900
14a	100	0.6	3	7	161	3	6	900
15a	100	0.8	4	9	65	4	9	900
1b	1	0.4	2	5	36	2	5	122
2b	1	0.6	3	9	10	3	9	88
3b	1	0.8	6	16	210	6	15	900
4b	10	0.4	2	5	59	2	5	281
5b	10	0.6	3	9	19	3	8	212
6b	10	0.8	6	13	277	6	13	900
7b	20	0.4	2	5	64	2	5	840
8b	20	0.6	3	9	28	3	8	280
9b	20	0.8	6	12	806	6	12	900
10b	50	0.4	2	5	80	2	5	900
11b	50	0.6	3	9	57	3	8	700
12b	50	0.8	6	12	422	6	12	900
13b	100	0.4	2	5	54	2	5	900
14b	100	0.6	3	9	74	3	8	900
15b	100	0.8	6	12	428	6	13	900
1c	1	0.4	2	6	12	2	6	96
2c	1	0.6	3	11	8	3	11	288
3c	1	0.8	6	16	40	6	16	181
4c	10	0.4	2	5	23	2	6	133
5c	10	0.6	3	7	12	3	7	277
6c	10	0.8	6	16	87	6	16	900
7c	20	0.4	2	5	23	2	6	355
8c	20	0.6	3	7	11	3	7	900
9c	20	0.8	6	16	126	6	16	900
10c	50	0.4	2	5	39	2	5	900
11c	50	0.6	3	7	19	3	7	900
12c	50	0.8	6	16	139	6	16	900
13c	100	0.4	2	5	40	2	5	900
14c	100	0.6	3	7	30	3	7	900
15c	100	0.8	6	16	211	6	15	900

Table 3.6: Results of larger problem instances with $f_i = 0 \; \forall \; i$ and TW1

Problem			Bounding schemes				Direct optimization			
n	h	α^O	LB	UB	G(UL)%	α^A	OBJ	α^A	G(CPX)%	G(OU)%
a1	50	0.267	8404	8404	0.00	0.510	8404	0.510	0.00	0.00
a2	100	0.267	8854	8854	0.00	0.510	8854	0.510	0.00	0.00
a3	200	0.267	9738	9738	0.00	0.483	9738	0.483	0.00	0.00
a4	50	0.400	8404	8404	0.00	0.510	8404	0.510	0.00	0.00
a5	100	0.400	8854	8854	0.00	0.510	8854	0.510	0.00	0.00
a6	200	0.400	9738	9738	0.00	0.483	9738	0.483	0.00	0.00
a7	50	0.534	8454	8454	0.00	0.541	8454	0.537	0.00	0.00
a8	100	0.534	8954	8954	0.00	0.541	8954	0.528*	0.00	0.00
a9	200	0.534	9954	9954	0.00	0.541	9954	0.532*	0.07	0.00
b1	50	0.305	8437	8437	0.00	0.542	8437	0.542	0.00	0.00
b2	100	0.305	9012	9012	0.00	0.531	9012	0.531	0.00	0.00
b3	200	0.305	10010	10010	0.00	0.488	10010	0.488	0.00	0.00
b4	50	0.458	8437	8437	0.00	0.542	8437	0.542	0.00	0.00
b5	100	0.458	9012	9012	0.00	0.531	9012	0.531	0.00	0.00
b6	200	0.458	10010	10010	0.00	0.488	10010	0.488	0.00	0.00
b7	50	0.610	8487	8487	0.00	0.613	8487	0.611	0.00	0.00
b8	100	0.610	9112	9212	1.09	0.622	9112	0.611	0.00	-1.10
b9	200	0.610	10258	10437	1.71	0.622	10312	0.611	0.00	-1.21
c1	50	0.371	7396	7396	0.00	0.522	7396	0.522	0.00	0.00
c2	100	0.371	7651	7651	0.00	0.479	7651	0.479	0.00	0.00
c3	200	0.371	8151	8151	0.00	0.479	8151	0.479	0.00	0.00
c4	50	0.557	7396	7446	0.67	0.583	7397	0.557	0.00	-0.68
c5	100	0.557	7746	7751	0.06	0.562	7747	0.557	0.00	-0.06
c6	200	0.557	8351	8351	0.00	0.580	8351	0.580	0.00	0.00
c7	50	0.742	7546	7546	0.00	0.751	7546	0.751	0.00	0.00
c8	100	0.742	7951	7999	0.59	0.743	7999	0.743	0.00	0.00
c9	200	0.742	8751	8899	1.65	0.743	8899	0.743	0.00	0.00
d1	50	0.234	13468	13468	0.00	0.450	13468	0.450	0.00	0.00
d2	100	0.234	14246	14246	0.00	0.427	14246	0.427	0.00	0.00
d3	200	0.234	15656	15656	0.00	0.384	15656	0.384	0.00	0.00
d4	50	0.350	13468	13468	0.00	0.450	13468	0.450	0.00	0.00
d5	100	0.350	14246	14246	0.00	0.427	14246	0.427	0.00	0.00
d6	200	0.350	15656	15656	0.00	0.384	15656	0.384	0.00	0.00
d7	50	0.467	13468	13518	0.37	0.469	13518	0.469	0.37	0.00
d8	100	0.467	14277	14377	0.70	0.469	14346	0.456*	0.70	-0.22
d9	200	0.467	15877	16077	1.24	0.469	16018	0.453*	1.73	-0.37
e1	50	0.362	10055	10055	0.00	0.592	10055	0.592	0.00	0.00
e2	100	0.362	10708	10708	0.00	0.587	10708	0.587	0.00	0.00
e3	200	0.362	11863	11863	0.00	0.561	11863	0.561	0.00	0.00
e4	50	0.543	10055	10055	0.00	0.592	10055	0.592	0.00	0.00
e5	100	0.543	10708	10708	0.00	0.587	10708	0.587	0.00	0.00
e6	200	0.543	11863	11863	0.00	0.561	11863	0.561	0.00	0.00
e7	50	0.724	10155	10205	0.49	0.730	10205	0.717*	0.48	0.00
e8	100	0.724	10908	11008	0.91	0.725	11008	0.715*	0.89	0.00
e9	200	0.724	12408	12511	0.83	0.724	12511	0.718*	0.39	0.00
f1	50	0.218	17040	17040	0.00	0.421	17040	0.421	0.00	0.00
f2	100	0.218	17891	17891	0.00	0.407	17891	0.407	0.00	0.00
f3	200	0.218	19480	19480	0.00	0.379	19480	0.379	0.00	0.00
f4	50	0.326	17040	17040	0.00	0.421	17040	0.421	0.00	0.00
f5	100	0.326	17891	17891	0.00	0.407	17891	0.407	0.00	0.00
f6	200	0.326	19480	19480	0.00	0.379	19480	0.379	0.00	0.00
f7	50	0.435	17040	17090	0.29	0.439	17090	0.435	0.29	0.00
f8	100	0.435	17891	18022	0.73	0.435	17923	0.425*	0.18	-0.55
f9	200	0.435	19491	19822	1.67	0.435	19691	0.424*	1.02	-0.67

Instance sizes: **a**→ 12 × 90, **b**→ 13 × 96, **c**→ 13 × 106, **d**→ 19 × 128, **e**→ 16 × 134, **f**→ 24 × 158

* infeasible solutions

Table 3.7: Results of larger problem instances with $f_i = 0 \; \forall \; i$ and TW1

Problem			Bounding schemes			Direct optimization		
n	h	α^O	$\sum Y_i$	$\sum S_i$	TIME	$\sum Y_i$	$\sum S_i$	TIME
a1	50	0.267	9	9	0	9	9	0
a2	100	0.267	9	9	0	9	9	0
a3	200	0.267	8	8	0	8	8	0
a4	50	0.400	9	9	0	9	9	0
a5	100	0.400	9	9	0	9	9	0
a6	200	0.400	8	8	0	8	8	0
a7	50	0.534	9	10	1	9	10	1800
a8	100	0.534	9	10	0	10	10	1800
a9	200	0.534	9	10	1	9	10	1800
b1	50	0.305	12	12	0	12	12	0
b2	100	0.305	11	11	0	11	11	0
b3	200	0.305	9	9	0	9	9	0
b4	50	0.458	12	12	0	12	12	0
b5	100	0.458	11	11	0	c	11	0
b6	200	0.458	9	9	0	9	9	0
b7	50	0.610	12	13	4	12	13	61
b8	100	0.610	11	13	5	11	12	123
b9	200	0.610	12	13	18	11	12	1800
c1	50	0.371	7	7	0	7	7	1
c2	100	0.371	5	5	0	5	5	1
c3	200	0.371	5	5	0	5	5	1
c4	50	0.557	8	8	17	7	7	150
c5	100	0.557	5	6	53	7	7	357
c6	200	0.557	5	6	108	5	6	59
c7	50	0.742	7	10	23	7	10	202
c8	100	0.742	6	9	115	6	9	341
c9	200	0.742	6	9	320	6	9	766
d1	50	0.234	17	17	0	17	17	1
d2	100	0.234	15	15	0	15	15	1
d3	200	0.234	12	12	0	12	12	0
d4	50	0.350	17	17	0	17	17	1
d5	100	0.350	15	15	0	15	15	1
d6	200	0.350	12	12	0	12	12	1
d7	50	0.467	17	18	2	17	18	1800
d8	100	0.467	16	17	5	15	16	1800
d9	200	0.467	16	17	13	17	17	1800
e1	50	0.362	14	14	0	14	14	1
e2	100	0.362	13	13	0	13	13	1
e3	200	0.362	11	11	0	11	11	1
e4	50	0.543	14	14	0	14	14	1
e5	100	0.543	13	13	0	13	13	1
e6	200	0.543	11	11	0	11	11	1
e7	50	0.724	14	17	54	15	17	1800
e8	100	0.724	13	16	42	14	16	1800
e9	200	0.724	12	15	65	12	15	1800
f1	50	0.218	19	19	0	19	19	1
f2	100	0.218	16	16	0	16	16	1
f3	200	0.218	14	14	0	14	14	1
f4	50	0.326	19	19	0	19	19	1
f5	100	0.326	16	16	0	16	16	1
f6	200	0.326	14	14	0	14	14	1
f7	50	0.435	19	20	3	20	20	1800
f8	100	0.435	17	18	6	17	17	1800
f9	200	0.435	17	18	9	17	17	1800

Instance sizes: **a**→ 12 × 90, **b**→ 13 × 96, **c**→ 13 × 106
d→ 19 × 128, **e**→ 16 × 134, **f**→ 24 × 158

Table 3.8: Results of larger problem instances with $f_i = 0 \; \forall \; i$ and TW2

Problem			Bounding schemes				Direct optimization			
n	h	α^O	LB	UB	G(UL)%	α^A	OBJ	α^A	G(CPX)%	G(OU)%
a1	50	0.385	8404	8404	0.00	0.513	8404	0.513	0.00	0.00
a2	100	0.385	8854	8854	0.00	0.513	8854	0.513	0.00	0.00
a3	200	0.385	9738	9738	0.00	0.486	9738	0.486	0.00	0.00
a4	50	0.578	8504	8504	0.00	0.580	8504	0.579	0.58	0.00
a5	100	0.578	9027	9054	0.30	0.579	9027	0.575*	0.81	-0.30
a6	200	0.578	9983	10154	1.69	0.584	10027	0.575*	0.43	-1.26
a7	50	0.770	9270	9307	0.40	0.771	9253	0.765*	0.30	-0.58
a8	100	0.770	10053	10179	1.24	0.771	10084	0.767*	0.35	-0.94
a9	200	0.770	11366	11486	1.05	0.771	11469	0.771	0.00	-0.14
b1	50	0.391	8437	8437	0.00	0.562	8437	0.562	0.00	0.00
b2	100	0.391	9012	9012	0.00	0.551	9012	0.551	0.00	0.00
b3	200	0.391	10010	10010	0.00	0.508	10010	0.508	0.00	0.00
b4	50	0.587	8437	8478	0.48	0.591	8437	0.587	0.00	-0.49
b5	100	0.587	9012	9037	0.27	0.587	9037	0.587	0.00	0.00
b6	200	0.587	10058	10220	1.59	0.589	10153	0.583*	0.11	-0.67
b7	50	0.782	8921	8991	0.78	0.782	8985	0.782	0.06	-0.07
b8	100	0.782	9811	9913	1.03	0.782	9894	0.782	0.00	-0.20
b9	200	0.782	11345	11440	0.83	0.783	11399	0.780*	0.25	-0.36
c1	50	0.400	7396	7396	0.00	0.551	7396	0.551	0.00	0.00
c2	100	0.400	7651	7651	0.00	0.498	7651	0.498	0.00	0.00
c3	200	0.400	8151	8151	0.00	0.498	8151	0.498	0.00	0.00
c4	50	0.600	7446	7446	0.00	0.614	7446	0.614	0.00	0.00
c5	100	0.600	7751	7751	0.00	0.602	7751	0.602	0.00	0.00
c6	200	0.600	8351	8351	0.00	0.602	8351	0.602	0.00	0.00
c7	50	0.800	7546	7546	0.00	0.800	7546	0.800	0.00	0.00
c8	100	0.800	8046	8046	0.00	0.800	8046	0.800	0.00	0.00
c9	200	0.800	8951	8951	0.00	0.806	8951	0.806	0.00	0.00
d1	50	0.400	13468	13468	0.00	0.496	13468	0.496	0.00	0.00
d2	100	0.400	14246	14246	0.00	0.474	14246	0.474	0.00	0.00
d3	200	0.400	15656	15656	0.00	0.429	15656	0.429	0.00	0.00
d4	50	0.600	13618	13718	0.73	0.605	13659	0.583*	0.86	-0.43
d5	100	0.600	14546	14818	1.84	0.601	14618	0.581*	1.17	-1.37
d6	200	0.600	16304	16723	2.51	0.600	16351	0.584*	1.35	-2.28
d7	50	0.800	15392	15497	0.68	0.800	15277	0.780*	1.21	-1.44
d8	100	0.800	16694	16781	0.52	0.800	16575	0.781*	1.68	-1.24
d9	200	0.800	18925	19089	0.86	0.800	18905	0.784*	1.93	-0.97
e1	50	0.400	10055	10055	0.00	0.621	10055	0.621	0.00	0.00
e2	100	0.400	10708	10708	0.00	0.615	10708	0.615	0.00	0.00
e3	200	0.400	11863	11863	0.00	0.588	11863	0.588	0.00	0.00
e4	50	0.600	10055	10055	0.00	0.621	10055	0.621	0.00	0.00
e5	100	0.600	10708	10708	0.00	0.615	10708	0.615	0.00	0.00
e6	200	0.600	11863	11905	0.35	0.601	11905	0.600	0.00	0.00
e7	50	0.800	10247	10255	0.08	0.805	10255	0.793*	0.00	0.00
e8	100	0.800	11097	11108	0.10	0.800	11108	0.800	0.00	0.00
e9	200	0.800	12705	12711	0.05	0.801	12711	0.800	0.00	0.00
f1	50	0.364	17040	17040	0.00	0.456	17040	0.456	0.00	0.00
f2	100	0.364	17891	17891	0.00	0.441	17891	0.441	0.00	0.00
f3	200	0.364	19480	19480	0.00	0.417	19480	0.417	0.00	0.00
f4	50	0.546	17185	17385	1.15	0.546	17240	0.537*	0.63	-0.84
f5	100	0.546	18235	18591	1.92	0.546	18291	0.535*	0.91	-1.65
f6	200	0.546	20137	20795	3.16	0.547	20238	0.537*	1.43	-2.75
f7	50	0.728	19222	19472	1.28	0.730	19130	0.715*	0.94	-1.79
f8	100	0.728	20698	20962	1.26	0.728	20724	0.715*	2.15	-1.15
f9	200	0.728	23181	23568	1.64	0.728	23284	0.718*	3.38	-1.22

Instance sizes: **a**→ 12 × 90, **b**→ 13 × 96, **c**→ 13 × 106, **d**→ 19 × 128, **e**→ 16 × 134, **f**→ 24 × 158

* infeasible solutions

Table 3.9: Results of larger problem instances with $f_i = 0 \; \forall \; i$ and TW2

Problem			Bounding schemes			Direct optimization		
n	h	α^O	$\sum Y_i$	$\sum S_i$	TIME	$\sum Y_i$	$\sum S_i$	TIME
a1	50	0.385	9	9	0	9	9	0
a2	100	0.385	9	9	0	9	9	0
a3	200	0.385	8	8	0	8	8	0
a4	50	0.578	10	11	9	10	11	1800
a5	100	0.578	10	11	8	10	10	1800
a6	200	0.578	10	11	10	10	10	1800
a7	50	0.770	10	18	64	10	17	1800
a8	100	0.770	9	15	68	10	15	1800
a9	200	0.770	9	12	48	9	12	1800
b1	50	0.391	12	12	0	12	12	0
b2	100	0.391	11	11	0	11	11	1
b3	200	0.391	9	9	0	9	9	0
b4	50	0.587	12	12	1	12	12	266
b5	100	0.587	12	12	2	12	12	1141
b6	200	0.587	11	11	23	11	11	1800
b7	50	0.782	12	19	207	12	20	1800
b8	100	0.782	12	17	71	12	17	1575
b9	200	0.782	11	14	106	11	14	1800
c1	50	0.400	7	7	0	7	7	1
c2	100	0.400	5	5	0	5	5	1
c3	200	0.400	5	5	0	5	5	1
c4	50	0.600	8	8	20	8	8	36
c5	100	0.600	5	6	64	5	6	120
c6	200	0.600	5	6	154	5	6	172
c7	50	0.800	7	10	73	7	10	444
c8	100	0.800	7	10	98	7	10	856
c9	200	0.800	5	9	27	5	9	441
d1	50	0.400	17	17	0	17	17	1
d2	100	0.400	15	15	0	15	15	1
d3	200	0.400	12	12	0	12	12	1
d4	50	0.600	18	22	36	17	20	1800
d5	100	0.600	16	21	18	17	20	1800
d6	200	0.600	15	18	85	16	17	1800
d7	50	0.800	18	29	1800	18	26	1800
d8	100	0.800	17	25	766	17	26	1800
d9	200	0.800	16	21	1248	17	22	1800
e1	50	0.400	14	14	0	14	14	1
e2	100	0.400	13	13	0	13	13	1
e3	200	0.400	11	11	0	11	11	2
e4	50	0.600	14	14	0	14	14	1
e5	100	0.600	13	13	0	13	13	1
e6	200	0.600	11	11	7	11	11	27
e7	50	0.800	14	18	93	15	18	1800
e8	100	0.800	13	17	55	13	17	834
e9	200	0.800	12	16	217	12	16	492
f1	50	0.364	19	19	0	19	19	1
f2	100	0.364	16	16	0	16	16	1
f3	200	0.364	14	14	0	14	14	1
f4	50	0.546	19	25	123	20	23	1800
f5	100	0.546	19	24	83	20	22	1800
f6	200	0.546	16	19	352	17	19	1800
f7	50	0.728	19	33	1800	20	32	1800
f8	100	0.728	19	29	1800	20	30	1800
f9	200	0.728	16	23	1800	18	24	1800

Instance sizes: **a**→ 12×90, **b**→ 13×96, **c**→ 13×106
d→ 19×128, **e**→ 16×134, **f**→ 24×158

Table 3.10: Results of larger problem instances with $f_i = 0 \ \forall \ i$ and TW3

Problem			Bounding schemes				Direct optimization			
n	h	α^O	LB	UB	G(UL)%	α^A	OBJ	α^A	G(CPX)%	G(OU)%
a1	50	0.400	8404	8404	0.00	0.671	8404	0.671	0.00	0.00
a2	100	0.400	8854	8854	0.00	0.671	8854	0.671	0.00	0.00
a3	200	0.400	9738	9738	0.00	0.650	9738	0.650	0.00	0.00
a4	50	0.600	8404	8404	0.00	0.671	8404	0.671	0.00	0.00
a5	100	0.600	8854	8854	0.00	0.671	8854	0.671	0.00	0.00
a6	200	0.600	9738	9738	0.00	0.650	9738	0.650	0.00	0.00
a7	50	0.800	8504	8504	0.00	0.819	8504	0.819	0.00	0.00
a8	100	0.800	9054	9054	0.00	0.819	9054	0.819	0.00	0.00
a9	200	0.800	10138	10138	0.00	0.810	10138	0.810	0.00	0.00
b1	50	0.400	8437	8437	0.00	0.649	8437	0.649	0.00	0.00
b2	100	0.400	9012	9012	0.00	0.641	9012	0.641	0.00	0.00
b3	200	0.400	10010	10010	0.00	0.608	10010	0.608	0.00	0.00
b4	50	0.600	8437	8437	0.00	0.649	8437	0.649	0.00	0.00
b5	100	0.600	9012	9012	0.00	0.641	9012	0.641	0.00	0.00
b6	200	0.600	10010	10010	0.00	0.608	10010	0.608	0.00	0.00
b7	50	0.800	8537	8537	0.00	0.830	8537	0.808	0.00	0.00
b8	100	0.800	9212	9212	0.00	0.826	9212	0.802	0.00	0.00
b9	200	0.800	10410	10410	0.00	0.811	10410	0.811	0.00	0.00
c1	50	0.400	7396	7396	0.00	0.567	7396	0.567	0.00	0.00
c2	100	0.400	7651	7651	0.00	0.514	7651	0.514	0.00	0.00
c3	200	0.400	8151	8151	0.00	0.514	8151	0.514	0.00	0.00
c4	50	0.600	7396	7396	0.00	0.600	7396	0.600	0.00	0.00
c5	100	0.600	7746	7746	0.00	0.600	7746	0.600	0.00	0.00
c6	200	0.600	8351	8351	0.00	0.615	8351	0.615	0.00	0.00
c7	50	0.800	7546	7546	0.00	0.810	7546	0.810	0.00	0.00
c8	100	0.800	7999	7999	0.00	0.802	7999	0.800	0.00	0.00
c9	200	0.800	8899	8899	0.00	0.802	8899	0.800	0.00	0.00
d1	50	0.400	13468	13468	0.00	0.631	13468	0.631	0.00	0.00
d2	100	0.400	14246	14246	0.00	0.608	14246	0.608	0.00	0.00
d3	200	0.400	15656	15656	0.00	0.574	15656	0.574	0.00	0.00
d4	50	0.600	13468	13468	0.00	0.631	13468	0.631	0.00	0.00
d5	100	0.600	14246	14246	0.00	0.608	14246	0.608	0.00	0.00
d6	200	0.600	15668	15668	0.00	0.601	15668	0.601	0.00	0.00
d7	50	0.800	13568	13568	0.00	0.800	13568	0.800	0.00	0.00
d8	100	0.800	14487	14487	0.00	0.800	14487	0.800	0.00	0.00
d9	200	0.800	16256	16256	0.00	0.809	16256	0.809	0.00	0.00
e1	50	0.400	10055	10055	0.00	0.641	10055	0.641	0.00	0.00
e2	100	0.400	10708	10708	0.00	0.638	10708	0.638	0.00	0.00
e3	200	0.400	11863	11863	0.00	0.619	11863	0.619	0.00	0.00
e4	50	0.600	10055	10055	0.00	0.641	10055	0.641	0.00	0.00
e5	100	0.600	10708	10708	0.00	0.638	10708	0.638	0.00	0.00
e6	200	0.600	11863	11863	0.00	0.619	11863	0.619	0.00	0.00
e7	50	0.800	10205	10205	0.00	0.809	10205	0.809	0.00	0.00
e8	100	0.800	11008	11008	0.00	0.807	11008	0.807	0.00	0.00
e9	200	0.800	12463	12463	0.00	0.800	12463	0.800	0.00	0.00
f1	50	0.400	17040	17040	0.00	0.596	17040	0.596	0.00	0.00
f2	100	0.400	17891	17891	0.00	0.586	17891	0.586	0.00	0.00
f3	200	0.400	19480	19480	0.00	0.563	19480	0.563	0.00	0.00
f4	50	0.600	17040	17040	0.00	0.600	17040	0.600	0.00	0.00
f5	100	0.600	17891	17891	0.00	0.600	17891	0.600	0.00	0.00
f6	200	0.600	19491	19491	0.00	0.600	19491	0.600	0.00	0.00
f7	50	0.800	17190	17190	0.00	0.801	17190	0.810	0.00	0.00
f8	100	0.800	18191	18191	0.00	0.804	18191	0.804	0.00	0.00
f9	200	0.800	20080	20080	0.00	0.800	20080	0.800	0.00	0.00

Instance sizes: **a**→ 12 × 90, **b**→ 13 × 96, **c**→ 13 × 106, **d**→ 19 × 128, **e**→ 16 × 134, **f**→ 24 × 158

* infeasible solutions

Table 3.11: Results of larger problem instances with $f_i = 0 \; \forall \; i$ and TW3

Problem			Bounding schemes			Direct optimization		
n	h	α^O	$\sum Y_i$	$\sum S_i$	TIME	$\sum Y_i$	$\sum S_i$	TIME
a1	50	0.400	9	9	0	9	9	0
a2	100	0.400	9	9	0	9	9	0
a3	200	0.400	8	8	0	8	8	0
a4	50	0.600	9	9	0	9	9	0
a5	100	0.600	9	9	0	9	9	1
a6	200	0.600	8	8	0	8	8	1
a7	50	0.800	9	11	1	9	11	6
a8	100	0.800	9	11	1	9	11	9
a9	200	0.800	8	10	1	8	10	18
b1	50	0.400	12	12	0	12	12	1
b2	100	0.400	11	11	0	11	11	1
b3	200	0.400	9	9	0	9	9	0
b4	50	0.600	12	12	0	12	12	1
b5	100	0.600	11	11	0	11	11	1
b6	200	0.600	9	9	0	9	9	0
b7	50	0.800	12	14	1	12	14	4
b8	100	0.800	11	13	1	11	13	5
b9	200	0.800	9	11	1	9	11	4
c1	50	0.400	7	7	0	7	7	1
c2	100	0.400	5	5	0	5	5	1
c3	200	0.400	5	5	0	5	5	1
c4	50	0.600	7	7	2	7	7	13
c5	100	0.600	7	7	13	7	7	113
c6	200	0.600	5	6	10	5	6	55
c7	50	0.800	7	10	10	7	10	137
c8	100	0.800	6	9	199	6	9	257
c9	200	0.800	6	9	386	6	9	243
d1	50	0.400	17	17	0	17	17	1
d2	100	0.400	15	15	0	15	15	1
d3	200	0.400	12	12	0	12	12	1
d4	50	0.600	17	17	0	17	17	1
d5	100	0.600	15	15	0	15	15	1
d6	200	0.600	13	13	1	13	13	3
d7	50	0.800	17	19	13	17	19	22
d8	100	0.800	16	18	4	16	18	53
d9	200	0.800	12	15	36	12	15	66
e1	50	0.400	14	14	0	14	14	1
e2	100	0.400	13	13	0	13	13	1
e3	200	0.400	11	11	0	11	11	1
e4	50	0.600	14	14	0	14	14	1
e5	100	0.600	13	13	0	13	13	1
e6	200	0.600	11	11	0	11	11	1
e7	50	0.800	14	17	7	14	17	202
e8	100	0.800	13	16	6	13	16	55
e9	200	0.800	11	14	183	11	14	304
f1	50	0.400	19	19	0	19	19	1
f2	100	0.400	16	16	0	16	16	1
f3	200	0.400	14	14	0	14	14	1
f4	50	0.600	19	19	1	19	19	4
f5	100	0.600	16	16	1	16	16	4
f6	200	0.600	16	16	6	16	16	22
f7	50	0.800	19	22	15	19	22	84
f8	100	0.800	16	19	10	16	19	83
f9	200	0.800	14	17	67	14	17	129

Instance sizes: **a**→ 12 × 90, **b**→ 13 × 96, **c**→ 13 × 106
d→ 19 × 128, **e**→ 16 × 134, **f**→ 24 × 158

Chapter 4

Multiple Products and Part Commonality

In this chapter, we extend our integrated modeling effort and solution methodology in the previous chapter to the single part, multi-product setting, where the part is inherently common across the products. As a side issue, we further investigate the benefits of exploiting part commonality within the integrated network design and inventory stocking problem.

4.1 Introduction

Having attacked the single-part problem with our approach in the previous chapter, we now investigate the problem with a single part common across multiple products while each product has a separate service level requirement. One way to solve this problem is to ignore part commonality and maintain a separate stock of the "common" part dedicated for each product. We can essentially decompose the problem across products by ignoring part commonality and each subproblem can

be solved independently. This is especially true when $f_i = 0 \ \forall \ i$, which leads to separate networks for individual parts. However, such a solution will be a costly solution due to overstocking. On the other hand, all products can share a single stock maintained at stocking facilities and may lead reduction in the overall stock (Collier 1981, 1982, Baker 1985, Baker et al. 1986). However, modeling and optimizing system costs and performance with shared stock levels and shared network is a much more difficult problem, but significant savings and better decision support may be obtained through exploiting commonality of parts and network. In this chapter, we show that we can handle issue of "sharing" stock very efficiently using the framework developed in the last chapter.

4.2 Modeling

We start with a model called RMP-PC (Part Commonality) that borrows notation from the earlier models in this dissertation and use an additional subscript p (in d_{jp}, X_{ijp} and θ_{ip}) to denote different products in a set P of products indexed by p. For example, d_{jp} denotes the mean annual demand of the part under consideration to be used to serve product p at demand point j. Similarly, X_{ijp} is 1 if the part's demand in product p is assigned to facility i, 0 otherwise. We now list the model:

$$\text{RMP-PC: Minimize} \ \sum_{i \in I} f_i Y_i + \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} t_{ij} d_{jp} X_{ijp} + \sum_{i \in I} h_i S_i \tag{4.1}$$

$$\text{s.t.} \quad \sum_{i \in I} X_{ijp} = 1 \ \forall \ j \in J, \ p \in P, \text{ and } d_{jp} > 0 \tag{4.2}$$

$$X_{ijp} \leq Y_i \ \forall \ i \in I, \ j \in J \text{ and } p \in P \tag{4.3}$$

$$\lambda_i = \tau \sum_{j \in J} \sum_{p \in P} d_{jp} X_{ijp} \; \forall \; i \in I \tag{4.4}$$

$$\sum_{i \in I} \sum_{j \in J} \delta_{ij} \theta_{ijp} \geq \alpha_p \tau \sum_{j \in J} d_{jp} \; \forall \; p \in P \tag{4.5}$$

$$\Theta_i \leq m_{kl} \lambda_i + b_{kl} + (s_{\max} - l) \sum_{n=1}^{r} (1 - e_{ln}) U_{in} \quad \forall \; l \in L, \; i \in I, \text{ and } k \in K \tag{4.6}$$

$$\theta_{ijp} \geq \tau d_{jp} (\beta_i + X_{ijp} - 1) \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.7}$$

$$\theta_{ijp} \leq \tau d_{jp} (\beta_i + 1 - X_{ijp}) \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.8}$$

$$\theta_{ijp} \leq \tau d_{jp} X_{ijp} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.9}$$

$$\Theta_i = \sum_{j \in J} \sum_{p \in P} \theta_{ijp} \; \forall \; i \in I \tag{4.10}$$

$$S_i = \sum_{n=1}^{r} 2^{n-1} U_{in} \; \forall \; i \in I \tag{4.11}$$

$$Y_i \leq \sum_{n=0}^{r} U_{in} \; \forall \; i \in I, \; j \in J \text{ and } p \in P \tag{4.12}$$

$$Y_i \geq U_{in} \; \forall \; i \in I \text{ and } n = 1, 2, 3, \ldots, r \tag{4.13}$$

$$\beta_i \in [0, 1] \; \forall \; i \in I \tag{4.14}$$

$$Y_i, U_{in}, X_{ijp} \in \{0, 1\} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.15}$$

$$\Theta_i, \theta_{ijp}, S_i, \lambda_i \geq 0 \; \forall \; i \in I, \; j \in J, \text{ and } p \in P. \tag{4.16}$$

In model RMP-PC, (4.4) and (4.5) are the only two constraint sets that need explanation. Constraints (4.4) are modified version of (3.35) and they pool the part's demand across multiple products (note an additional summation over $p \in P$). It is essential to notice that and S_i and $\beta_i \; \forall \; i$ do not have new subscript p on them since all products share the common stock of the part stocked at any facility and hence use the same fill rate. Constraints (4.5) represent multiple service level constraints, one for each product.

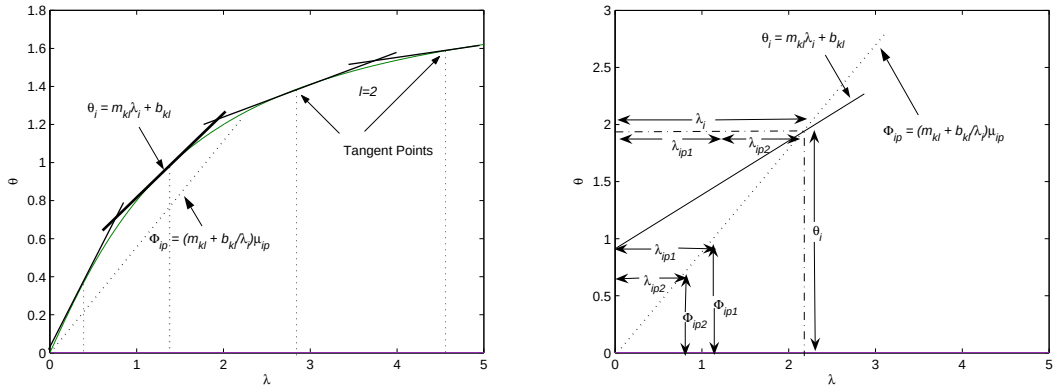


Figure 4.1: Decomposition of Θ_i over multiple products: Θ_i is a linear function of λ_i, Φ_{ip} is also linear function of μ_{ip} and passes through origin, the two lines intersect at (λ_i, Θ_i). In the figure above, we have the following relations: $\Theta_i = \Phi_{ip_1} + \Phi_{ip_2}$, $\lambda_i = \mu_{ip_1} + \mu_{ip_2}$, $\Phi_{ip} = \beta_i \mu_{ip}$, and $\beta_i = m_{kl} + b_{kl}/\lambda_i$

.

4.3 Solution Methodology

The RPM-PC model along with the outer-approximation mechanism can be used to obtain a near-optimal solution to the problem (with multiple products and part commonality). However, RPM-PC as a monolithic model is now a larger and more difficult MIP, especially with the increasing number products. To solve the problem more efficiently, we propose lower and upper bounding schemes for the RMP-PC model, which are similar to the ones developed for the single-part, single-product setting in the previous chapter.

4.3.1 Lower Bounding

For lower bounding, we again make use of dependency relaxation discussed in section 3.4.1. Therefore, instead of (4.4), we use the following modified definition of $\lambda_i \ \forall \ i$ (ignoring the contribution of the demands coming from outside the service time-

window of facility i):

$$\lambda_i = \tau \sum_{j \in J} \sum_{p \in P} \delta_{ij} d_{jp} X_{ijp} \; i \in I. \tag{4.17}$$

Additionally, we define two new variables μ_{ip} and Φ_{ip} as follows:

$$\mu_{ip} = \tau \sum_{j \in J} \delta_{ij} d_{jp} X_{ijp} \; \forall \; i \in I \text{ and } p \in P \tag{4.18}$$

$$\Phi_{ip} = \beta_i \mu_{ip} \; \forall \; i \in I \text{ and } p \in P. \tag{4.19}$$

At each i, μ_{ip} is the portion of λ_i consisting of the demand for the part only for product p:

$$\sum_{p \in P} \mu_{ip} = \lambda_i \; \forall \; i \in I. \tag{4.20}$$

Similarly, Φ_{ip} is the portion of Θ_i that is due to product p. Hence, under dependency relaxation, we have

$$\sum_{p \in P} \Phi_{ip} = \Theta_i \; \forall \; i \in I, \tag{4.21}$$

or, incorporating the fill rates, we have

$$\sum_{p \in P} \beta_i \mu_{ip} = \beta_i \lambda_i \; \forall \; i \in I. \tag{4.22}$$

In other words, variable Φ_{ip} is the portion of Θ_i that must be "reserved" at facility i to achieve the system-wide service level requirement for product p. Note the similarity in (3.10) and (4.19). However, despite the similarity in conceptualization and mathematical expressions of the single-product Θ_i and the multi-product Φ_{ip}, the two functions have different properties. Given values of S_i, the single-product Θ_i is a concave function of λ_i, but the multi-product Φ_{ip} is not a concave function of μ_{ip}, because Φ_{ip} is also function of the mean lead time demands due to products

other than p, i.e., μ_{ip} also depends on $\mu_{ip'}$, $\forall\ p' \in P$ and $p' \neq p$. To see this, we note that the total demand for the part at facility i is an aggregation of all individual product-level demands, and the overall fill rate at the facility, as a function of the total demand (and the stock level), in turn affects all product-level mean lead time demands satisfied directly from stock, μ_{ip}'s. However, we know that the multi-product, aggregate Θ_i is still a concave function of aggregate mean lead time demand, λ_i. The main issue here is to be able to derive a computationally efficient way to distribute the value of Θ_i into product-level Φ_{ip}'s, which are not necessarily concave functions of their demand variables, μ_{ip}'s. Using variables Φ_{ip}, we write the service level constraints as follows:

$$\sum_{i \in I} \Phi_{ip} \geq \alpha_p \tau \sum_{j \in J} d_{jp} \ \forall\ p \in P. \tag{4.23}$$

To be able to explicitly use Φ_{ip} in our model, we must handle nonlinearity in (4.19). Since the tangent lines to the g-curve at are of the form $m_{kl}\lambda + b_{kl}$, with slopes m_{kl} and intercepts b_{kl} (refer to Section 3.3.3). We use the fact that given i: Θ_i, β_i, and λ_i are connected by a linear relation and for some $k \in K$ the following holds:

$$m_{kl}\lambda_i + b_{kl} \quad < \quad m_{k'l}\lambda_i + b_{k'l} \ \forall\ k' \in K,\ k' \neq k, \text{ and } l \in L \tag{4.24}$$

Hence the only tangential constraint corresponding to kth point will be binding and the following holds:

$$\Theta_i = \beta_i \lambda_i = m_{kl}\lambda_i + b_{kl} \quad \Rightarrow \quad \beta_i = m_{kl} + \frac{b_{kl}}{\lambda_i} \ \forall\ l \in L. \tag{4.25}$$

Using (4.19) we can write:

$$\Phi_{ip} = m_{kl}\mu_{ip} + b_{kl}\frac{\mu_{ip}}{\lambda_i} \quad \forall\ l \in L \text{ and } p \in P. \tag{4.26}$$

Further we replace the ratio $\frac{\mu_{ip}}{\lambda_i}\ \forall\ i \in I$ and $p \in P$ with a single variable R_{ip} that has the following properties:

$$\begin{aligned} \sum_{p \in P} R_{ip} &= 1\ \forall\ i \in I & (4.27)\\ R_{ip} &\in [0,1]\ \forall\ i \in I \text{ and } p \in P. & (4.28) \end{aligned}$$

Finally, using (4.26) and R_{ip} we can represent (4.19) in the following manner:

$$\begin{aligned} \Phi_{ip} \leq\ & m_{kl}\mu_{ip} + b_{kl}R_{ip} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{in} \\ & \forall\ l \in L,\ i \in I,\ p \in P,\ k \in K. \end{aligned} \tag{4.29}$$

Note that (4.29) is a set of linear constraints very similar to (4.6). These constraints are remarkable in purposes they serve: outer-approximation, distribution of Θ_i across multiple ps as Φ_{ip}, and also determination the stock level. With this development we write the lower bounding model RMP-PC-LB as follows:

$$\text{RMP-PC-LB: Minimize } \sum_{i \in I} f_i Y_i + \sum_{i \in I}\sum_{j \in J}\sum_{p \in P} t_{ij} d_{jp} X_{ijp} + \sum_{i \in I} h_i S_i \tag{4.30}$$

$$\begin{aligned} \text{s.t.} \quad \sum_{i \in I} X_{ijp} &= 1\ \forall\ j \in J,\ p \in P,\ \text{s.t.: } d_{jp} > 0 & (4.31)\\ X_{ijp} &\leq Y_i\ \forall\ i \in I,\ j \in J, \text{ and } p \in P & (4.32)\\ \mu_{ip} &= \tau \sum_{j \in J} \delta_{ij} d_{jp} X_{ijp}\ \forall\ i \in I \text{ and } p \in P & (4.33) \end{aligned}$$

$$\sum_{i \in I} \Phi_{ip} \geq \alpha_p \tau \sum_{j \in J} d_{jp} \; \forall \; p \in P \tag{4.34}$$

$$\Phi_{ip} \leq m_{kl}\mu_{ip} + b_{kl}R_{ip} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{in} \quad \forall \; l \in L, \; i \in I, \; p \in P, \text{ and } k \in K \tag{4.35}$$

$$\sum_{p \in P} R_{ip} = 1 \; \forall \; i \in I \tag{4.36}$$

$$S_i = \sum_{n=1}^{r} 2^{n-1} U_{in} \; \forall \; i \in I \tag{4.37}$$

$$Y_i \leq \sum_{n=0}^{r} U_{in} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.38}$$

$$Y_i \geq U_{in} \; \forall \; i \in I \text{ and } n = 1, 2, 3, \ldots, r \tag{4.39}$$

$$R_{ip} \in [0, 1] \; \forall \; i \in I \text{ and } p \in P \tag{4.40}$$

$$\Phi_{ip}, \mu_{ip}, S_i \geq 0 \; \forall \; i \in I \text{ and } p \in P \tag{4.41}$$

$$Y_i, U_{in}, X_{ijp} \in \{0, 1\} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P. \tag{4.42}$$

The objective function and the rest of the constraints are the same as before. RMP-PC-LB is a much simpler model as compared to RMP-PC, since dependency relaxation eliminates the need of explicit consideration of θ_{ijp} and associated constraints (4.7)-(4.9) for linearization. Further introduction of Φ_{ip}, μ_{ip} and R_{ip} variables replaces Θ_i and λ_i in the RMP-PC model. More importantly, R_{ip} variables accurately represent the ratio μ_{ip}/λ_i, making the model an MIP.

4.3.2 Upper Bounding

The process of computing upper bounds with multiple products and part commonality is the same as explained in Section 3.4.2, except that at every iteration, we now test the feasibility of service level constraints for all products $p \in P$ and boost individual α_p if needed. Also, due to the requirement of handling multiple α values

simultaneously, we do not perform any bisection search to tighten upper bounds after the first feasible solution is obtained. However, we try to conservatively boost α_p's in very small amounts, so that the first feasible solution is obtained without much overshoot in α_p values. with Below is the listing of the upper bounding algorithm:

Upper Bounding Algorithm

1. Set $u = 0$, pick the value of $\gamma \in (0, 1)$, and set $\alpha_p := \alpha_p^O \ \forall \ p \in P$.
2. Solve RMP-PC-LB and set $u := u + 1$.
3. Calculate α_p^A for the solution obtained.
4. If $\alpha_p^A \geq \alpha_p^O$, $\forall \ p \in P$ then STOP, and report the feasible solution. Otherwise,
   (a) Boost α_p for products that are infeasible in service: Set $\alpha_p := \min\{\alpha_p^M, \alpha_p + \gamma(\alpha_p^O - \alpha_p^A)\}$, $\forall \ p \in P$ such that $\alpha_p^A < \alpha_p^O$.
   (b) Go to Step 2.

4.3.3 Ignoring Part Commonality

Using the same notation as in RMP-PC-LB below we write RMP-IGN-LB model that ignores the part commonality. In a way, this model assumes that at every facility a separate stock of part is maintained for each product.

$$\text{RMP-IGN-LB: Minimize} \quad \sum_{i \in I} f_i Y_i + \sum_{i \in I} \sum_{j \in J} \sum_{p \in P} t_{ij} d_{jp} X_{ijp} + \sum_{i \in I} \sum_{p \in P} h_i S_{ip} \tag{4.43}$$

$$\text{s.t.} \quad \sum_{i \in I} X_{ijp} = 1 \ \forall \ j \in J, \ p \in P, \ \text{s.t.: } d_{jp} > 0 \tag{4.44}$$

$$X_{ijp} \leq Y_i \ \forall \ i \in I, \ j \in J, \ \text{and } p \in P \tag{4.45}$$

$$\mu_{ip} = \tau \sum_{j \in J} \delta_{ij} d_{jp} X_{ijp} \; \forall \; i \in I \text{ and } p \in P \tag{4.46}$$

$$\sum_{i \in I} \Phi_{ip} \geq \alpha_p \tau \sum_{j \in J} d_{jp} \; \forall \; p \in P \tag{4.47}$$

$$\Phi_{ip} \leq m_{kl} \mu_{ip} + b_{kl} + (s_{\max} - l) \sum_{n=1}^{r} (1 - e_{ln}) U_{inp} \quad \forall \; l \in L, \; i \in I, \; p \in P, \text{ and } k \in K \tag{4.48}$$

$$S_{ip} = \sum_{n=1}^{r} 2^{n-1} U_{inp} \; \forall \; i \in I \tag{4.49}$$

$$Y_i \leq \sum_{n=0}^{r} U_{inp} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P \tag{4.50}$$

$$Y_i \geq U_{inp} \; \forall \; i \in I \text{ and } n = 1, 2, 3, \ldots, r \tag{4.51}$$

$$\Phi_{ip}, \mu_{ip}, S_{ip} \geq 0 \; \forall \; i \in I \text{ and } p \in P \tag{4.52}$$

$$Y_i, U_{inp}, X_{ijp} \in \{0, 1\} \; \forall \; i \in I, \; j \in J, \text{ and } p \in P. \tag{4.53}$$

RMP-IGN-LB is rather straightforward to explain. We use an additional subscript 'p' in S_{ip} and U_{inp} to distinguish stocks for products. This causes the solution to keep separate stocks for each product hence ignore part commonality across products. Everything else remains the same. The solution of RMP-IGN-LB gives us lower bound on optimal cost, however, RMP-IGN-LB along with upper bounding α boosting algorithm can also be used to compute upper bounds.

4.4 Computational Study

To study the strength of our bounding schemes, we do not compare the bounds with direct optimization as we have done in the previous chapter. Direct optimization for multi-product problems is shown to be further inferior (as compared to its performance for the single-product setting) in our preliminary experiments. However, we compare two settings with each other ("consider" vs. "ignore" part commonality)

and compare the heuristic solutions' upper bounds with our dependency relaxation lower bounds.

4.4.1 Test Problem Data

The data of three networks used for the comparison is provided by a SPL group at a large computer hardware manufacturer. We have three networks, with varying sizes: 25 candidate locations and 159 customers (or, 25×159 for short), 19×128, and 13×96. The first two networks have three products (and a single part), and the last network has two products using a single part. Four different values of fixed facility cost (f_i) are tested, $0, 10, 100$ and 1000 for all i. Holding cost (h_i) is fixed at 65. Customer demands for the part for each product are generated using uniform random numbers between 1 and 3. Transportation costs and time-window defining data (δ_{ij}'s) are given in the real data. The data have three service time-windows (representing 2 hr, 4 hr, and 12 hr response times), which are used here separately for each instance, to analyze the effects of the time-window on our results.

The lead time τ is 7 days for all facilities, and the maximum allowed stock level s_{max} is 5. We initialize the outer-approximation of the graphs of Θ_i with 13 tangent lines at the breakpoints of the mean lead time demand that correspond to the percentages of the maximum possible mean lead time demand for a facility. The maximum mean lead time demand is $\Lambda = \tau \sum_{p \in P} \sum_{j \in J} d_{jp}$ and demands at the breakpoints are computed using the following breakpoints:

$$k \in \{.05, 0.10, 0.15, 0.20, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}.$$

To solve MIPs we use CPLEX 9.0 installed on a PC with dual Xeon 1.8 GHz processor and 1 GB RAM running Suse Linux operating system. Due to the presence

of multiple products and part commonality, the integrated problem is now much harder to solve. Even the lower bounding problem can be a challenge to solve to optimality. Hence, to be able to solve the lower bounding problem efficiently for the "part commonality consideration" case, the integrality restriction on X_{ijp} variables is relaxed. To perform more conservative updates on α_p's, the boosting factor γ is kept at 0.5 (as opposed to 1 in the single product case). Also, in the multi-product setting, it is not possible to perform bisection search for boosted values of α_p's. Hence, we stop the boosting with the first confirmed feasible solution. The time limit of 150 seconds is used for every time a lower bounding model is solved, either during outer-approximation or during α-boosting.

4.4.2 Results and Discussion

Tables 3.6-3.11 present results obtained using RPM-PC-LB and compare them with those obtained using RMP-IGN-LB. For each time-window, we present results for 36 problem instances generated using combinations of 3 networks, 3 service levels, and 4 levels of fixed costs $(0, 10, 100, 1000)$.

The computational results for 3 time-windows (TW1, TW2, and TW3) are organized using 6 tables, 2 for each time-window. The structure of these tables is slightly different from those used in the previous chapter. In the first table for each time-window, we do not report α_p^A values here since all upper bounding solutions are always feasible across all products. Lower bounds reported for the "part commonality considered" case (LB_1) correspond to the values obtained with continuous X_{ijp}'s. Upper bounds (UB_1 for the part commonality case and UB_2 for the ignored part commonality) correspond to binary X_{ijp}'s so that all final feasible solutions have each customer's demand for a product assigned to a single facility. As before

G(UL)% is the percentage difference between the corresponding lower and upper bounds. In the G(BC)% column, we report the percentage gap between the two different upper bound values obtained for two cases: (1) X_{ijp} are defined binary, and (2) X_{ijp} are defined continuous between 0 and 1. Also, the G(COM)% column reports the percentage difference between the upper bounds of the two part commonality considerations, and defined as $100(UB_2 - UB_1)/UB_1$. In a way, this column shows the percentage increase in total costs due to ignoring part commonality.

In the second table for each time-window, along with the network specific data, we report the total number of open facilities ($\sum Y_i$) and total stock level across all locations ($\sum S_i$) in the optimal solutions, for both cases of part commonality considerations. As before, we report solution times in column (T). As the part commonality consideration is poised to affect the inventory decisions the most, we also report the inventory cost component of the objective function (INV_1 for part commonality, INV_2 for no part commonality), and show the percentage difference between them in column G(INV)%, which is defined as $100(INV_2 - INV_1)/INV_1$.

Tables 4.1 and 4.2 compare the case "consider part commonality" with "ignore part commonality" for TW1. The values in the G(UL)% column suggest that our bounding schemes can be used effectively to solve the problem with multiple products and part commonality considerations. The average G(UL)% gap across 36 instances is 0.70% and maximum gap is 4.60%. Even better UB-LB gaps are reported for the model that ignores part commonality. As expected, the values in G(COM)% are all positive, which suggests that considering part commonality may result in significant savings (up to 13%) in total costs (comparing upper bounds). The results also show that the amount of savings due to part commonality (or increase in costs if common parts are modeled independently) can be quite different

across different networks, but given a network, the savings depends heavily on fixed costs. With larger f_i (as opening and operating a facility becomes more expensive), the percentage savings of explicit modeling of part commonality decrease. Note that we keep the unit holding cost and unit transportation cost the same across the instances. Hence, we believe that the interaction between these parameters affect the overall savings, which are all captured under changing fixed cost values. Moreover, we observe that the UB of part commonality considered is even lower than the LB of part commonality ignored.

To see how much of the total cost savings is due to inventory, we compare overall savings with savings in inventory costs (i.e., compare G(COM)% in Table 4.1 and G(INV)% in Table 4.2). Savings in inventory costs can be as high as 150%, but this saving is always offset by other network costs because the networks decisions (location/allocation decisions and stock levels) are also different, which is an interesting observation. As expected, ignoring part commonality results in greater total stock at smaller number of open facilities when compared to the solutions with explicit part commonality consideration. This observation reduces to the "stock more at less number of places" strategy of ignoring part commonality and the "stock less at more number of places" strategy of considering part commonality. It is needless to say, but part commonality when considered in an integrated problem has an effect on not only inventory decisions but also network decisions.

We make one more observation here on the differences in CPU times for solving the problems with and without part commonality. In general, ignoring part commonality reduces the complexity of the problem greatly and the integrated model becomes easier to solve. Moreover, when f_i is set to 0 for all i (or small enough to be negligible), ignoring part commonality enables us to decompose the

integrated problem across products, solve the subproblems independently for each product (similar to the ones, the single-product, single-part problems, attacked in the previous chapter) and combine the product-level solutions to obtain an optimal integrated solution. Even when the fixed facility costs are positive (preventing the full product-based decomposition), the ignore part commonality problems have reasonable computation times. However, solving the integrated problem with part commonality consideration can be challenging in terms of solution time (note the CPU times in Table 4.4). The integrated problem with part commonality is indeed a very hard problem to solve, and our computational experience suggests that good solutions can be obtained relatively quickly if integrality requirement of X_{ijp} variables is relaxed (although still longer than the ones with part commonality ignored). The values in G(BC)% column suggest that the solutions obtained by such a relaxation are very close to those with obtained with binary restrictions.

Similarly, Tables 4.3-4.4 and Tables 4.5-4.6 compare the case "consider part commonality" with "ignore part commonality" for TW2 and TW3 respectively. A quick comparison of G(UL)% values and solution times across time-windows suggests that instances in TW2 are the hardest) and in TW3 are the easiest, however, the effects of part commonality on network design and inventory decisions are consistent across time-windows. Most values in G(BC)% column for TW3 are 0 since larger time-windows makes the problem easier and model gains the standard UFL-like properties.

4.5 Summary

In this chapter, we presented an approach to model part commonalities in the integrated model of network design and inventory stocking, especially extending our

θ-based outer-approximation and dependency relaxation-based approach designed originally for the single-part, single-product setting in the previous chapter. We show that our approach extends to the multi-product setting with little modification in the lost-sales fill rate case, leading to the mathematical models that are still largely tractable for reasonable size problems. Considering part commonality is desirable due to potential savings in total costs, but it makes the integrated problem much harder to solve to optimality. This chapter's computational results based on multi-product problems show the strength and robustness of the modeling and solution methodology originally developed for the single-product setting. We observe that inventory savings due to part commonality can be significant, some of which is offset by the increase in other costs such as facility costs and transportation costs. Overall, considering part commonality can bring up to 13% savings in total costs. In the next chapter, we give further insights regarding the integrated problem to show the usefulness of the overall approach in even larger perspective.

Table 4.1: Effects of Part Commonality on Integrated Model's Total Costs (TW1)

Problem			Part Commonality Considered				Part Commonality Ignored			
n	s	f	LB_1	UB_1	G(UL)%	G(BC)%	LB_2	UB_2	G(UL)%	G(COM)%
g1	0.4	0	16764	16764	0.00	0.00	18935	18935	0.00	12.95
g2	0.6	0	16764	16764	0.00	0.00	18935	18935	0.00	12.95
g3	0.8	0	16764	16959	1.16	0.00	19065	19130	0.34	12.80
g4	0.4	10	16954	16954	0.00	0.00	19095	19095	0.00	12.63
g5	0.6	10	16954	16954	0.00	0.00	19095	19095	0.00	12.63
g6	0.8	10	16954	17084	0.77	-0.38	19225	19290	0.34	12.92
g7	0.4	100	18455	18455	0.00	0.00	20379	20379	0.00	10.42
g8	0.6	100	18455	18455	0.00	0.00	20436	20444	0.04	10.78
g9	0.8	100	18505	18765	1.41	-0.03	20665	20730	0.31	10.47
g10	0.4	1000	26325	26455	0.49	0.00	27235	27300	0.24	3.19
g11	0.6	1000	27898	27963	0.23	-0.23	29068	29079	0.04	3.99
g12	0.8	1000	30966	31899	3.01	0.20	32721	32916	0.60	3.19
h1	0.4	0	13987	13987	0.00	0.00	15776	15776	0.00	12.79
h2	0.6	0	13987	13987	0.00	0.00	15811	15841	0.19	13.26
h3	0.8	0	13987	14182	1.39	0.46	16006	16109	0.65	13.59
h4	0.4	10	14157	14157	0.00	0.00	15874	15874	0.00	12.13
h5	0.6	10	14157	14157	0.00	0.00	15918	15942	0.15	12.61
h6	0.8	10	14157	14417	1.84	0.90	16126	16215	0.55	12.47
h7	0.4	100	15441	15441	0.00	0.00	16675	16675	0.00	7.99
h8	0.6	100	15445	15571	0.81	0.00	16749	16749	0.00	7.57
h9	0.8	100	15559	15819	1.67	0.09	17136	17266	0.76	9.14
h10	0.4	1000	20967	21097	0.62	0.00	21617	21617	0.00	2.46
h11	0.6	1000	21930	22546	2.81	0.00	22710	23326	2.72	3.46
h12	0.8	1000	24867	26011	4.60	0.00	26232	26362	0.50	1.35
i1	0.4	0	8913	8913	0.00	0.00	9690	9690	0.00	8.71
i2	0.6	0	8913	8913	0.00	0.00	9690	9690	0.00	8.71
i3	0.8	0	8978	9043	0.72	0.00	9758	9758	0.00	7.91
i4	0.4	10	9033	9033	0.00	0.00	9796	9796	0.00	8.45
i5	0.6	10	9033	9033	0.00	0.00	9796	9796	0.00	8.45
i6	0.8	10	9098	9163	0.71	0.00	9875	9940	0.66	8.47
i7	0.4	100	10046	10046	0.00	0.00	10650	10650	0.00	6.01
i8	0.6	100	10046	10046	0.00	0.00	10650	10650	0.00	6.01
i9	0.8	100	10150	10270	1.18	0.26	10826	10826	0.00	5.42
i10	0.4	1000	16150	16192	0.26	0.26	16540	16540	0.00	2.15
i11	0.6	1000	16772	16837	0.39	0.00	17227	17292	0.38	2.70
i12	0.8	1000	17610	17805	1.11	0.00	18130	18130	0.00	1.83

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Table 4.2: Effects of Part Commonality on Network Design and Inventory Decisions (TW1)

Problem			Part Commonality Considered				Part Commonality Ignored				
n	s	f	$\sum Y_i$	$\sum S_i$	INV_1	T	$\sum Y_i$	$\sum S_i$	INV_2	T	G(INV)%
g1	0.4	0	19	19	1235	7	16	48	3120	1	152.63
g2	0.6	0	19	19	1235	8	16	48	3120	1	152.63
g3	0.8	0	19	22	1430	21	16	51	3315	2	131.82
g4	0.4	10	19	19	1235	7	16	48	3120	1	152.63
g5	0.6	10	19	19	1235	8	16	48	3120	1	152.63
g6	0.8	10	19	21	1365	23	16	51	3315	3	142.86
g7	0.4	100	16	16	1040	7	12	36	2340	1	125.00
g8	0.6	100	16	16	1040	7	12	37	2405	1	131.25
g9	0.8	100	17	21	1365	16	16	51	3315	8	142.86
g10	0.4	1000	7	9	585	53	7	22	1430	3	144.44
g11	0.6	1000	9	16	1040	50	9	33	2145	15	106.25
g12	0.8	1000	14	24	1560	340	13	51	3315	7	112.50
h1	0.4	0	17	17	1105	4	11	33	2145	1	94.12
h2	0.6	0	17	17	1105	4	11	34	2210	1	100.00
h3	0.8	0	17	20	1300	11	14	45	2925	20	125.00
h4	0.4	10	17	17	1105	4	9	27	1755	1	58.82
h5	0.6	10	17	17	1105	4	10	31	2015	1	82.35
h6	0.8	10	17	21	1365	13	13	42	2730	3	100.00
h7	0.4	100	12	12	780	4	8	24	1560	1	100.00
h8	0.6	100	12	14	910	6	9	28	1820	1	100.00
h9	0.8	100	14	19	1235	319	11	39	2535	4	105.26
h10	0.4	1000	5	8	520	13	5	16	1040	1	100.00
h11	0.6	1000	7	13	845	37	7	25	1625	1	92.31
h12	0.8	1000	11	20	1300	48	10	39	2535	3	95.00
i1	0.4	0	12	12	780	2	11	22	1430	0	83.33
i2	0.6	0	12	12	780	2	11	22	1430	0	83.33
i3	0.8	0	12	14	910	452	12	25	1625	0	78.57
i4	0.4	10	12	12	780	2	10	20	1300	0	66.67
i5	0.6	10	12	12	780	2	10	20	1300	0	66.67
i6	0.8	10	12	14	910	452	11	24	1560	1	71.43
i7	0.4	100	10	10	650	2	9	18	1170	0	80.00
i8	0.6	100	10	10	650	3	9	18	1170	0	80.00
i9	0.8	100	11	14	910	454	10	22	1430	1	57.14
i10	0.4	1000	6	6	390	6	6	12	780	0	100.00
i11	0.6	1000	7	9	585	12	7	16	1040	1	77.78
i12	0.8	1000	8	15	975	456	8	20	1300	3	33.33

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Table 4.3: Effects of Part Commonality on Integrated Model's Total Costs (TW2)

Problem			Part Commonality Considered				Part Commonality Ignored			
n	s	f	LB_1	UB_1	G(UL)%	G(BC)%	LB_2	UB_2	G(UL)%	G(COM)%
g1	0.4	0	16764	16764	0.00	0.00	18935	18935	0.00	12.95
g2	0.6	0	17376	17497	0.70	-0.32	19575	19671	0.49	12.42
g3	0.8	0	18728	19154	2.27	0.97	21153	21284	0.62	11.12
g4	0.4	10	16954	16954	0.00	0.00	19095	19095	0.00	12.63
g5	0.6	10	17566	17695	0.73	-0.04	19735	19831	0.49	12.08
g6	0.8	10	18914	19337	2.24	0.92	21323	21437	0.54	10.86
g7	0.4	100	18455	18455	0.00	0.00	20387	20387	0.00	10.47
g8	0.6	100	19067	19294	1.19	-0.01	21142	21279	0.65	10.29
g9	0.8	100	20515	21238	3.52	1.50	22747	22824	0.34	7.47
g10	0.4	1000	26617	27107	1.84	-0.19	27223	27528	1.12	1.55
g11	0.6	1000	28657	29162	1.76	-0.06	29587	29779	0.65	2.12
g12	0.8	1000	31231	31953	2.31	-0.26	32482	32640	0.49	2.15
h1	0.4	0	13987	13987	0.00	0.00	15776	15776	0.00	12.79
h2	0.6	0	14552	14786	1.61	0.09	16595	16713	0.71	13.03
h3	0.8	0	15977	16322	2.16	0.57	18123	18261	0.76	11.88
h4	0.4	10	14157	14157	0.00	0.00	15883	15886	0.02	12.21
h5	0.6	10	14731	15020	1.96	0.34	16722	16848	0.76	12.17
h6	0.8	10	16157	16531	2.31	0.67	18266	18344	0.43	10.97
h7	0.4	100	15441	15571	0.84	0.25	16749	16814	0.39	7.98
h8	0.6	100	16134	16379	1.52	0.07	17729	17834	0.59	8.88
h9	0.8	100	17630	18016	2.19	0.92	19374	19445	0.36	7.93
h10	0.4	1000	20849	21735	4.25	0.22	21416	22131	3.34	1.82
h11	0.6	1000	23461	23918	1.95	-0.95	24260	24373	0.47	1.90
h12	0.8	1000	26930	27304	1.39	-0.53	28190	28322	0.47	3.73
i1	0.4	0	8913	8913	0.00	0.00	9690	9690	0.00	8.71
i2	0.6	0	9151	9194	0.47	0.04	9941	9960	0.20	8.34
i3	0.8	0	10025	10168	1.43	1.01	10820	10823	0.03	6.44
i4	0.4	10	9033	9033	0.00	0.00	9796	9796	0.00	8.45
i5	0.6	10	9271	9306	0.37	-0.09	10061	10074	0.14	8.26
i6	0.8	10	10145	10272	1.25	0.85	10938	10946	0.07	6.56
i7	0.4	100	10046	10046	0.00	0.00	10650	10650	0.00	6.01
i8	0.6	100	10323	10380	0.56	0.14	11051	11082	0.28	6.76
i9	0.8	100	11204	11315	0.99	0.49	11928	12041	0.94	6.42
i10	0.4	1000	16591	16670	0.48	0.24	16948	16954	0.03	1.70
i11	0.6	1000	17398	17499	0.58	-0.13	17793	17793	0.00	1.68
i12	0.8	1000	18411	18467	0.31	0.10	18807	18815	0.04	1.88

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Table 4.4: Effects of Part Commonality on Network Design and Inventory Decisions (TW2)

Problem			Part Commonality Considered				Part Commonality Ignored				
n	s	f	$\sum Y_i$	$\sum S_i$	INV_1	T	$\sum Y_i$	$\sum S_i$	INV_2	T	G(INV)%
g1	0.4	0	19	19	1235	9	16	48	3120	1	152.63
g2	0.6	0	19	23	1495	183	16	53	3445	5	130.43
g3	0.8	0	19	34	2210	1456	17	61	3965	1110	79.41
g4	0.4	10	19	19	1235	9	16	48	3120	1	152.63
g5	0.6	10	19	23	1495	304	16	53	3445	6	130.43
g6	0.8	10	19	33	2145	1455	17	62	4030	957	87.88
g7	0.4	100	16	16	1040	9	13	39	2535	2	143.75
g8	0.6	100	16	22	1430	432	15	51	3315	7	131.82
g9	0.8	100	18	35	2275	1455	15	56	3640	1152	60.00
g10	0.4	1000	6	13	845	382	5	21	1365	18	61.54
g11	0.6	1000	7	17	1105	512	7	30	1950	311	76.47
g12	0.8	1000	10	28	1820	582	9	45	2925	98	60.71
h1	0.4	0	17	17	1105	5	11	33	2145	1	94.12
h2	0.6	0	18	21	1365	388	13	43	2795	42	104.76
h3	0.8	0	18	29	1885	555	15	55	3575	234	89.66
h4	0.4	10	17	17	1105	5	11	33	2145	2	94.12
h5	0.6	10	17	23	1495	340	12	41	2665	13	78.26
h6	0.8	10	18	31	2015	697	14	50	3250	226	61.29
h7	0.4	100	12	14	910	6	9	29	1885	3	107.14
h8	0.6	100	14	18	1170	474	11	38	2470	8	111.11
h9	0.8	100	15	25	1625	582	12	45	2925	185	80.00
h10	0.4	1000	5	10	650	293	5	20	1300	2	100.00
h11	0.6	1000	6	15	975	221	6	27	1755	14	80.00
h12	0.8	1000	9	21	1365	635	9	42	2730	159	100.00
i1	0.4	0	12	12	780	3	11	22	1430	0	83.33
i2	0.6	0	12	14	910	206	12	26	1690	15	85.71
i3	0.8	0	12	18	1170	494	12	29	1885	156	61.11
i4	0.4	10	12	12	780	3	10	20	1300	0	66.67
i5	0.6	10	12	14	910	154	11	25	1625	11	78.57
i6	0.8	10	12	18	1170	800	12	29	1885	167	61.11
i7	0.4	100	10	10	650	3	9	18	1170	0	80.00
i8	0.6	100	11	13	845	455	11	25	1625	2	92.31
i9	0.8	100	12	17	1105	496	10	26	1690	163	52.94
i10	0.4	1000	5	7	455	8	5	12	780	1	71.43
i11	0.6	1000	6	10	650	437	6	15	975	452	50.00
i12	0.8	1000	6	13	845	473	6	18	1170	154	38.46

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Table 4.5: Effects of Part Commonality on Integrated Model's Total Costs (TW3)

Problem			Part Commonality Considered				Part Commonality Ignored			
n	s	f	LB_1	UB_1	G(UL)%	G(BC)%	LB_2	UB_2	G(UL)%	G(COM)%
g1	0.4	0	16764	16764	0.00	0.00	18935	18935	0.00	12.95
g2	0.6	0	16764	16764	0.00	-0.78	18935	18935	0.00	12.95
g3	0.8	0	16959	16959	0.00	0.00	19130	19130	0.00	12.80
g4	0.4	10	16954	16954	0.00	0.00	19095	19095	0.00	12.63
g5	0.6	10	16954	16954	0.00	-0.77	19095	19095	0.00	12.63
g6	0.8	10	17149	17149	0.00	0.00	19290	19290	0.00	12.49
g7	0.4	100	18455	18455	0.00	0.00	20379	20379	0.00	10.42
g8	0.6	100	18455	18520	0.35	-0.35	20379	20379	0.00	10.04
g9	0.8	100	18650	18650	0.00	-0.18	20631	20639	0.04	10.66
g10	0.4	1000	24197	24197	0.00	0.00	24327	24327	0.00	0.54
g11	0.6	1000	24262	24327	0.27	0.00	24522	24522	0.00	0.80
g12	0.8	1000	24457	24457	0.00	0.00	24717	24717	0.00	1.06
h1	0.4	0	13987	13987	0.00	0.00	15776	15776	0.00	12.79
h2	0.6	0	13987	13987	0.00	0.00	15776	15892	0.73	13.62
h3	0.8	0	14117	14182	0.46	0.00	15971	16006	0.22	12.86
h4	0.4	10	14157	14157	0.00	0.00	15874	15874	0.00	12.13
h5	0.6	10	14157	14157	0.00	0.00	15874	15874	0.00	12.13
h6	0.8	10	14287	14352	0.45	0.00	16078	16134	0.35	12.41
h7	0.4	100	15441	15441	0.00	0.00	16675	16675	0.00	7.99
h8	0.6	100	15445	15510	0.42	0.00	16675	16675	0.00	7.51
h9	0.8	100	15636	15640	0.03	0.03	16935	16935	0.00	8.28
h10	0.4	1000	19778	19843	0.33	0.33	19973	19973	0.00	0.66
h11	0.6	1000	19908	19908	0.00	-0.33	20038	20166	0.64	1.30
h12	0.8	1000	20038	20038	0.00	0.00	20363	20363	0.00	1.62
i1	0.4	0	8913	8913	0.00	0.00	9690	9690	0.00	8.71
i2	0.6	0	8913	8913	0.00	0.00	9690	9690	0.00	8.71
i3	0.8	0	9043	9043	0.00	0.00	9820	9820	0.00	8.59
i4	0.4	10	9033	9033	0.00	0.00	9796	9796	0.00	8.45
i5	0.6	10	9033	9033	0.00	0.00	9796	9796	0.00	8.45
i6	0.8	10	9163	9163	0.00	0.00	9926	9926	0.00	8.33
i7	0.4	100	10046	10046	0.00	0.00	10650	10650	0.00	6.01
i8	0.6	100	10046	10046	0.00	0.00	10650	10650	0.00	6.01
i9	0.8	100	10176	10176	0.00	0.00	10780	10780	0.00	5.93
i10	0.4	1000	15666	15731	0.41	0.00	15926	15926	0.00	1.24
i11	0.6	1000	15731	15731	0.00	0.00	15926	15926	0.00	1.24
i12	0.8	1000	15796	15861	0.41	0.00	16121	16121	0.00	1.64

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Table 4.6: Effects of Part Commonality on Network Design and Inventory Decisions (TW3)

Problem			Part Commonality Considered				Part Commonality Ignored				
n	s	f	$\sum Y_i$	$\sum S_i$	INV_1	T	$\sum Y_i$	$\sum S_i$	INV_2	T	G(INV)%
g1	0.4	0	19	19	1235	13	16	48	3120	1	152.63
g2	0.6	0	19	19	1235	21	16	48	3120	1	152.63
g3	0.8	0	19	22	1430	278	16	51	3315	16	131.82
g4	0.4	10	19	19	1235	13	16	48	3120	1	152.63
g5	0.6	10	19	19	1235	30	16	48	3120	1	152.63
g6	0.8	10	19	21	1430	292	16	51	3315	6	131.82
g7	0.4	100	16	16	1040	13	12	36	2340	1	125.00
g8	0.6	100	16	16	1105	25	12	36	2340	1	111.76
g9	0.8	100	17	21	1235	130	12	40	2600	20	110.53
g10	0.4	1000	7	9	260	27	2	6	390	2	50.00
g11	0.6	1000	9	16	390	44	2	9	585	3	50.00
g12	0.8	1000	14	24	520	150	2	12	780	163	50.00
h1	0.4	0	17	17	1105	7	11	33	2145	1	94.12
h2	0.6	0	17	17	1105	10	11	33	2145	1	94.12
h3	0.8	0	17	20	1300	514	12	39	2535	14	95.00
h4	0.4	10	17	17	1105	7	9	27	1755	1	58.82
h5	0.6	10	17	17	1105	8	9	27	1755	1	58.82
h6	0.8	10	17	21	1300	465	9	31	2015	164	55.00
h7	0.4	100	12	12	780	7	8	24	1560	1	100.00
h8	0.6	100	12	14	910	22	8	24	1560	1	71.43
h9	0.8	100	14	19	1040	649	8	28	1820	6	75.00
h10	0.4	1000	5	8	260	13	2	6	390	1	50.00
h11	0.6	1000	7	13	325	19	2	7	455	1	40.00
h12	0.8	1000	11	20	455	30	2	12	780	155	71.43
i1	0.4	0	12	12	780	3	11	22	1430	0	83.33
i2	0.6	0	12	12	780	3	11	22	1430	0	83.33
i3	0.8	0	12	14	910	158	11	24	1560	1	71.43
i4	0.4	10	12	12	780	3	10	20	1300	0	66.67
i5	0.6	10	12	12	780	3	10	20	1300	0	66.67
i6	0.8	10	12	14	910	308	10	22	1430	1	57.14
i7	0.4	100	10	10	650	3	9	18	1170	0	80.00
i8	0.6	100	10	10	650	3	9	18	1170	0	80.00
i9	0.8	100	11	14	780	13	9	20	1300	0	66.67
i10	0.4	1000	6	6	390	4	4	9	585	0	50.00
i11	0.6	1000	7	9	390	4	4	9	585	0	50.00
i12	0.8	1000	8	15	520	7	4	12	780	0	50.00

Instance sizes: **g**→ $25 \times 159 \times 3$, **h**→ $19 \times 128 \times 3$, **i**→ $13 \times 96 \times 2$

Chapter 5

Further Insights and Extensions

This chapter gives more insight into the problem structure and broadens the perspective introduced in the earlier chapters. First, we discuss our understanding of the poor performance of direct optimization of Mixed Integer Nonlinear Programming (MINLP) version of the integrated problem using MINLP solvers – especially having shown the convexity. Then, we discuss several relaxation schemes one can try to compute lower bounds on the integrated problem. We then uncover some special cases of the general integrated problem we attack in the overall dissertation, and identify ways to deal with them more efficiently. Finally, we discuss extensions or generalizations of the integrated model and comment on general difficulty of these problems.

5.1 MINLP Solver's Dilemma

The concavity of the g functions is useful for convexification and outer-approximation scheme used so far, but it could also be advantageous for solving the original problem directly as a Mixed Integer Nonlinear Programming (MINLP) model. In fact, we

attempted this using the commercially available DICOPT solver (Duran and Grossmann 1986, Grossmann et al. 2001) that tries to solve the overall model as a series of MIP and NLPs solved successively. The MIP is a relaxed master program which optimizes over a polyhedron made of tangents which encompasses the original polyhedron, while the NLP is a subproblem which is solved to generate more tangential cutting planes (a.k.a. feasibility or optimality cuts) to either reject or correct the latest solution obtained by solving master MIP. We experienced a few challenges: First, the solver's decomposition scheme led to a quite large master MIP problems. Second, since the scheme stops with reporting a solution only when a network obtained by master MIP is "accepted" (feasible) by the NLP subproblem. When the master MIP proposes an infeasible network, the NLP subproblem passes cuts to the master to change the network. Often, it takes large number of cuts before the master MIP changes the network. In some cases, the MINLP solver ends up trying many candidate networks which are rejected continuously. Moreover, the iterations in the overall MINLP scheme are very costly in terms of both computation time and memory, due to the growing size of MIP over iterations.

5.2 More Relaxation Schemes

In this section, we discuss two more lower bounding schemes which do not necessarily provide us stronger lower bounds than our dependency relaxation, yet they provide us meaningful insights into the problem structure. Using the notation defined for a single part and single product formulation introduced before, we present two more relaxation schemes.

5.2.1 Unit Fill Rate

With this, we assume that $\beta_i = 1$ for all i regardless of the stock and demand levels at the facilities. Below is the formulation of such a relaxation, denoted by MLOC-UFR:

$$\text{MLOC-UFR: Minimize} \quad \sum_{i \in I}(f_i + h_i)Y_i + \sum_{i \in I}\sum_{j \in J} t_{ij} d_j X_{ij} \tag{5.1}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{i \in I} X_{ij} &= 1 \ \forall \ j \in J && (5.2)\\
X_{ij} &\leq Y_i \ \forall \ i \in I \text{ and } j \in J && (5.3)\\
\sum_{i \in I}\sum_{j \in J} \delta_{ij} d_j X_{ij} &\geq \alpha \sum_{j \in J} d_j && (5.4)\\
Y_i, \ X_{ij} &\in \{0,1\} \ \forall \ i \in I \text{ and } j \in J. && (5.5)
\end{aligned}$$

MLOC-UFR is a relatively simple model, obtained by modifying the UFL problem with the additional consideration of the service level constraint and the inventory term in the objective function. In fact, (5.4) is just a demand coverage constraint. MLOC-UFR, however, does not completely ignore the inventory but stocks just 1 unit of the part at each open facility. SPL systems with extremely low demand can take advantage of this model's simplicity as in reality no more than one unit is stocked at any facility and once stocked the facility can provide practically 100% fill rate, which may make the relaxation close to the actual, more restrictive problem.

5.2.2 Time-Window Relaxation

The time-window relaxation scheme is very similar to dependency relaxation. Under this relaxation, we set all δ_{ij} values to 1. This simplifies the RMP models greatly, leading to the model denoted by TWR-LB. TWR-LB is different from the dependency relaxation lower bounding model LBP only in one set of constraints, (5.10):

$$\text{TWR-LB: Minimize } \sum_{i\in I} f_i Y_i + \sum_{i\in I}\sum_{j\in J} t_{ij} d_j X_{ij} + \sum_{i\in I} h_i S_i \tag{5.6}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{i\in I} X_{ij} &= 1 \ \forall\ j \in J && (5.7)\\
X_{ij} &\le Y_i \ \forall\ i \in I \text{ and } j \in J && (5.8)\\
\sum_{i\in I} \Theta_i &\ge \tau\alpha \sum_{j\in J} d_j && (5.9)\\
\lambda_i &= \tau \sum_{j\in J} d_j X_{ij} \ \forall\ i \in I && (5.10)\\
\Theta_i &\le m_{ks}\lambda_i + b_{ks} + (s_{\max} - l)\sum_{n=1}^{r}(1-e_n)U_{in} \\
&\quad \forall\ l \in L,\ i \in I,\ \text{and } k \in K && (5.11)\\
S_i &= \sum_{n=1}^{r} 2^{n-1} U_{in} \ \forall\ i \in I && (5.12)\\
Y_i &\le \sum_{n=0}^{r} U_{in} \ \forall\ i \in I && (5.13)\\
Y_i &\ge U_{in} \ \forall\ i \in I \text{ and } n = 1, 2, 3, \ldots, r && (5.14)\\
U_{in}, X_{ij} &\in \{0,1\} \ \forall\ i \in I \text{ and } j \in J && (5.15)\\
\beta_i, Y_i &\in [0,1] \ \forall\ i \in I. && (5.16)
\end{aligned}$$

A comparison between time-window relaxation and dependency relaxation gives the insight that the core difficulty of the problem indeed lies in δ_{ij} values. Setting all $\delta_{ij} = 1$ essentially means that the "radius" in which a facility can serve within the time-window is infinite (or a large value such that every customer is within the time-window of every facility). The actual value of this radius determines δ_{ij} values; the less the value of the radius is, the more difficult the problem is. In fact, a radius smaller than a threshold in the extreme may even make the problem infeasible (no matter how the network designed or stocked it is impossible to provide the service required within the time-window). As compared to dependency relaxation, the time-window relaxation is the other extreme that makes the problem very easy. In between the two extremes there is varying levels of difficulty of the integrated problem.

5.3 Special Cases

We now discuss several special cases of the integrated problem for which the modeling can be simplified greatly. All special cases are illustrated and discussed for the single-part, single-product setting.

1. **Case 1:** The two-stage solution technique, however, shown to be a heuristic for the integrated problem, can provide an optimal solution at a cheaper computational cost in a special case. Such a case arises when only one facility is open in the solution thus obtained must be optimal to the integrated problem. It is because when only one facility is stocked – we achieve maximum pooling of stocks, so the inventory cost is at the minimum and since joint cost of transportation and fixed cost of opening is already minimized by the first stage, hence this solution must be a minimum cost (the optimal) solution.

2. **Case 2:** A special case of the integrated problem is obtained when the individual part fill rates at open facilities are fixed to a constant value less than unity. One such candidate value the target service level itself, i.e., $\beta_i = \alpha$ for all open facilities, however, it would require us to cover 100% demand within specified time-window. Under this scenario, the integrated problem can be solved in two consecutive stages, as discussed in section 2.2 by using $\beta_i = \alpha \ \forall \ i$ instead of $\beta_i = 1 \ \forall \ i$ in the location stage. The second stage is just a table lookup problem for each i with given $\beta_i = \alpha$ and λ_i.

3. **Case 3:** Another special case is when we just need to decide "stock one unit or not" at each open facility. Again, the problem can be simplified, leading to a model called INT-SO1, which is illustrated below:

$$\text{INT-S01: Minimize} \quad \sum_{i \in I}(f_i + h_i)Y_i + \sum_{i \in I}\sum_{j \in J} t_{ij} d_j X_{ij} \tag{5.17}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{i \in I} X_{ij} &= 1 \ \forall \ j \in J && (5.18)\\
X_{ij} &\leq Y_i && (5.19)\\
\sum_{i \in I}\sum_{j \in J} \delta_{ij}\theta_{ij} &\geq \tau\alpha \sum_{j \in J} d_j && (5.20)\\
\lambda_i &= \tau \sum_{j \in J} d_j X_{ij} \ \forall \ i \in I && (5.21)\\
\theta_{ij} &\geq \tau d_j(\beta_i + X_{ij} - 1) \ \forall \ i \in I \text{ and } j \in J && (5.22)\\
\theta_{ij} &\leq \tau d_j(\beta_i + 1 - X_{ij}) \ \forall \ i \in I \text{ and } j \in J && (5.23)\\
\theta_{ij} &\leq \tau d_j X_{ij} \ \forall \ i \in I \text{ and } j \in J && (5.24)\\
\sum_{j \in J} \theta_{ij} &\leq m_k \lambda_i + b_k \ \forall \ i \in I \text{ and } k \in K && (5.25)\\
X_{ij} &\in \{0, 1\} \ \forall \ i \in I \text{ and } j \in J && (5.26)\\
\beta_i, Y_i &\in [0, 1] \ \forall \ i \in I. && (5.27)
\end{aligned}$$

Note that the INT-S01 model is different from the unit fill rate relaxation. MLOC-UFR is a lower bounding model, while INT-S01 is an exact model for the case when $S_i = 1$ for all open facilities and uses the exact values of fill rates at each open facility. INT-S01 model is especially useful for low-demand SPL systems with a very high inventory stocking cost. However, this restricted model may be infeasible with higher demands.

5.4 Extensions

The two main extensions we consider are regarding the part and product diversity. Modeling and solution methodology based on the concavity of g function and outer-approximation can be easily extended to multiple parts, products and their part commonality structure. Borrowing notation from the earlier models, and using an additional subscript 'c' to distinguish variables across different parts (or *components*), we list two major modeling extensions focusing on the corresponding lower bounding models.

5.4.1 Multiple Parts and Single Product

First, we consider the single product requiring a single time-based service level to be satisfied across multiple parts that go into the product. We denote the corresponding lower bounding problem as MPSP-LB and list its formulation below:

$$\text{MPSP-LB: Minimize} \quad \sum_{i \in I} f_i Y_i + \sum_{i \in I} \sum_{j \in J} \sum_{c \in C} t_{ij} d_{cj} X_{cij} + \sum_{i \in I} \sum_{c \in C} h_{ci} S_{ci} \qquad (5.28)$$

$$
\begin{aligned}
\text{s.t.} \sum_{i\in I} X_{cij} &= 1 \ \forall\ c \in C \text{ and } j \in J && (5.29)\\
X_{cij} &\leq Y_i \ \forall\ c \in C,\ i \in I,\ \text{and } j \in J && (5.30)\\
\sum_{c\in C}\sum_{i\in I} \Theta_{ci} &\geq \tau \sum_{c\in C}\sum_{j\in J} \alpha d_{cj} && (5.31)\\
\lambda_{ci} &= \tau \sum_{j\in J} \delta_{ij} d_{cj} X_{cij} \ \forall\ c \in C \text{ and } i \in I && (5.32)\\
\Theta_{ci} &\leq m_{ckl}\lambda_{ci} + b_{ckl} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{cin} \\
&\quad \forall\ c \in C,\ l \in L,\ i \in I,\ \text{and } k \in K && (5.33)\\
S_{ci} &= \sum_{n=1}^{r} 2^{n-1} U_{cin} \ \forall\ c \in C \text{ and } i \in I && (5.34)\\
X_{cij} &\leq \sum_{n=0}^{r} U_{cin} \ \forall\ c \in C,\ i \in I,\ \text{and } j \in J && (5.35)\\
Y_i &\geq U_{cin} \ \forall\ c \in C,\ i \in I,\ \text{and } n = 1, 2, 3, \ldots, r && (5.36)\\
Y_i, U_{cin}, X_{cij} &\in \{0, 1\} \ \forall\ c \in C,\ i \in I,\ \text{and } j \in J && (5.37)\\
\beta_{ci} &\in [0, 1] \ \forall\ c \in C \text{ and } i \in I && (5.38)\\
S_{ci}, \lambda_{ci} &\geq 0 \ \forall\ c \in C \text{ and } i \in I. && (5.39)
\end{aligned}
$$

Note that here we suppress subscript "p" as it is not needed. Furthermore, since we have only product, part commonality need not be considered. The rest of the formulation is straightforward.

5.4.2 Multiple Parts and Multiple Products

As an ultimate extension of the models introduced in this dissertation, we now consider the multi-part, multi-product scenario. The associated lower bounding problem is denoted by MPMP-LB, as listed below:

$$\text{MPMP-LB: Minimize } \sum_{i\in I} f_i Y_i + \sum_{i\in I}\sum_{j\in J}\sum_{c\in C}\sum_{p\in P} t_{ij} d_{cjp} X_{cijp} + \sum_{i\in I}\sum_{c\in C} h_{ci} S_{ci} \quad (5.40)$$

$$\begin{aligned}
\text{s.t. } \sum_{i\in I} X_{cijp} &= 1 \ \forall\ c \in C,\ j \in J,\ \text{and } p \in P && (5.41)\\
X_{cijp} &\le Y_i \ \forall\ c \in C,\ i \in I,\ j \in J \text{ and } p \in P && (5.42)\\
\sum_{c\in C}\sum_{i\in I} \Theta_{ci} &\ge \tau \sum_{c\in C}\sum_{j\in J}\sum_{p\in P} \alpha_p d_{cjp} && (5.43)\\
\lambda_{ci} &= \tau \sum_{j\in J}\sum_{p\in P} \delta_{ij} d_{cjp} X_{cijp} \ \forall\ c \in C \text{ and } i \in I && (5.44)\\
\Theta_{ci} &\le m_{ckl}\lambda_{ci} + b_{ckl} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{cin} \\
&\quad \forall\ c \in C,\ l \in L,\ i \in I,\ \text{and } k \in K && (5.45)\\
S_{ci} &= \sum_{n=1}^{r} 2^{n-1} U_{cin} \ \forall\ c \in C \text{ and } i \in I && (5.46)\\
X_{cijp} &\le \sum_{n=0}^{r} U_{cin} \ \forall\ c \in C,\ i \in I,\ j \in J,\ \text{and } p \in P && (5.47)\\
Y_i &\ge U_{cin} \ \forall\ c \in C,\ i \in I,\ \text{and } n = 1, 2, 3, \ldots, r && (5.48)\\
Y_i, U_{cin}, X_{cijp} &\in \{0, 1\} \ \forall\ c \in C,\ i \in I,\ j \in J,\ \text{and } p \in P && (5.49)\\
\beta_{ci} &\in [0, 1] \ \forall\ c \in C \text{ and } i \in I && (5.50)\\
S_{ci}, \lambda_{ci} &\ge 0 \ \forall\ c \in C \text{ and } i \in I. && (5.51)
\end{aligned}$$

Model MPMP-LB can be used to compute lower bounds for an integrated problem with multiple parts and multiple products with explicit part commonality consideration. Note that constraints (5.43) are slightly different (an additional summation over $c \in C$ in the left hand side) as the same product-based service level can be achieved through multiple parts. A summation over $p \in P$ in (5.44) indicates

the part commonality consideration.

In the two models above, the objective function terms and the rest of the constraints are self explanatory. These models along with α-boosting scheme can be used to compute upper bounds for the corresponding problems. The associated models for direct optimization and the models that ignore part commonality are rather straightforward to formulate and are not provided here for brevity.

Our experimental computation with these models both with direct optimization and with our own outer-approximation/α-boosting approach has not been very satisfactory, especially in terms of solution quality. It is conceivable that in presence of multiple parts, dependency relaxation and the α-boosting algorithm are not the best alternatives due to multiplicity of relaxation and boosting. In this regard, rigorous related computational investigation and further development are left for future research.

5.5 Summary

In this chapter, we discussed the our insight we gain in the problem structure while we developed the bounding schemes and solution technique reported. The special cases we covered simplify the problem greatly, and whenever possible existence of such cases must be explored before we attempt to solve the problem using the full scale models and solution techniques. It is entirely possible that under certain scenario fill rates are not modeled as variables but as parameters or we just have to make "stock a unit or not" decisions along with network decisions. The extensions we discuss in this chapter are perhaps naïve suggestions to start modeling the problem in a larger perspective. We provide more discussion on such extensions in the later half of the next chapter.

Chapter 6

Conclusions and Future Research

6.1 Summary

In this dissertation, we investigate an integrated network design and inventory stocking problem, especially designed for low-demand systems such as service parts logistics networks. Due to crucial interactions between network design and inventory decisions caused by time-based service levels requirements that exist in such systems, the two sets of decisions must be considered simultaneously for overall optimization of the system costs and service. Depending upon assumptions and requirements, modeling and solution development for an integrated problem could be challenging. First we show that concurrent consideration of network design and inventory decisions in a single model leads to a large scale mixed integer nonlinear programming (MINLP) formulation. The main challenge is to handle fill rate computation which plays a central role in optimization. We then introduce a piece-wise linearization

scheme for fill rate approximation to linearize the MINLP to a MIP, but the MIP model can be prohibitively large and ultimately still difficult to solve. To overcome this difficulty, one may try different and potentially more efficient approaches to approximate fill rates. Such approaches often depend upon specific properties of fill rate functions and service level constraints. For the special case of lost-sales fill rates, which is commonly used in SPL systems, we introduce a new decision variable, and propose an outer-approximation based modeling and solution scheme which makes use of the concavity of the left hand side of the service level constraint in terms of the new variable. Along with an exact solution scheme which becomes possible with the full linearization of the model in which the nonlinearities are handled much more efficiently than the straightforward fill rate approximation, based on the same model, we introduce a totally new relaxation based on breaking the dependency between the mean lead time demand variable and the demand that may come outside the time-window at each facility. This not only results in an effective lower bounding mechanism for the integrated problem, but also opens a way to an upper bounding heuristic in which a parametric version of the lower bounding model is solved iteratively, where the required service level serving as the parameter of the problem is boosted so that the optimal solution satisfies the original service level targets. We show that the overall approach is quite effective and works flawlessly in the single part and single product setting. To the best of our knowledge, models integrating the two sets of decisions with variable fill rates do not exist in the literature, except a few conducted by Kutanoglu et al. (Candas and Kutanoglu 2006a,b, Iyoob and Kutanoglu 2006). We also address consideration of part commonality issues in the integrated network design and inventory stocking problem. Through a detailed computational study at every step of the way, we show the strength of

our modeling and solution techniques. With our computational experiences on variety of problem settings based on both randomly generated and industry-specific instances, we make the following important observations:

- Our results are based on $s_{\max} = 5$, which is justifiably sufficient for SPL systems. In our computational experience, none of the open facilities ever stocked 5 units of stock. This means $s_{\max}$ is not used to represent capacity of an open facility, but just a loose upper bound on how much a facility can stock which can be obtained prior to optimization. This is necessary for computational reasons as the model stipulates integer stock levels. The model, in fact, can be used for any value of $s_{\max}$ with the caution that the number of constraints will grow linearly, and the number of binary variables will grow only logarithmically (base 2). If a fine distinction between $S = l$ and $S = l+1$ for any l is not important or significant then one can use the scaled version of stocking variables in the model and reduce the number of binary variables further.

- Dependency relaxation makes the model easier to solve, which suggests that perhaps the core difficulty of the integrated problem lies in δ_{ij} values. This is indeed true because the fill rate at facility i is affected by the aggregated mean lead time demand $\lambda_i = \lambda_i^{\text{in}} + \lambda_i^{\text{out}}$, while contribution to the service level is achieved by only λ_i^{in}. In the original problem (and in direct optimization), we must differentiate between λ_i^{in} and λ_i^{out}. Under dependency relaxation we ignore λ_i^{out}, which eliminates nonlinearity in the substitution scheme, making the model easier to solve. The only nonlinearity remaining is in the calculation of Θ_i which is handled efficiently by the outer-approximation scheme.

- In all of our formulations X_{ij} can be relaxed to take real values in between 0

and 1. Depending on what is required in the actual problem, one can use the overall approach to solve either problem, with the proper restrictions on X_{ij}'s. This is surely a desirable feature of the proposed model and the associated solution methodology.

6.2 Future Work

The research work reported in this dissertation is only a first-step contribution towards the larger goal of integrating supply chain decisions and system-wide optimization. We leave out considerable amount of work for future research. Below is a nonexhaustive list of potential research items that are of future concerns:

6.2.1 Lower Bounding Problem

As illustrated already in the previous chapters, dependency relaxation simplifies the RMP models significantly, and in fact, only this way, we are able to solve the integrated problem using the introduced bounding schemes. However, various lower bounding dependency relaxation models (LBP, RMP-PC-LB, RMP-IGN-LB, and MPSP-LB, MPMP-LB) can still be very large and difficult MIPs. A potential research area is to investigate the structure in the dependency relaxation model and develop solution schemes to solve it more efficiently. In this regard, our limited exploration using standard ideas such as Lagrangian relaxation, Lagrangian decomposition, and column generation schemes etc. have not shown any promising direction. The main difficulty here is that the dependency relaxation model must be solved either to optimality, or must be relaxed further. We cannot use any heuristic technique to solve the dependency relaxation problem to obtain lower bounds. Further relaxing the lower bounding problem may lead to difficulty in developing good

upper bounds using the α-boosting scheme and may widen the G(UL)% values. One idea that seemed to work in the multi-product setting is to relax the integrality of X_{ijp} variables, without causing deterioration on the upper bound values. In some settings, even the problems with relaxed X_{ij}'s are hard, hence a more creative ideas are needed.

6.2.2 Multiple Time Windows

All the models considered in this dissertation have only one time-window, over which a product's achieved time-based service must be at least a target level. We consider multiple time-windows in the computational experiments, but they are considered separately as a controlled experimental factor. However, real SPL systems typically involve multiple hourly time-windows over which increasing service levels are required over longer time-windows. Considering multiple time-windows and associated service level requirements in a single formulation is an interesting and challenging problem. In terms of modeling, large part of the model will remain the same with this extension, except that the service level constraints must be written for every window. A generic form of service level constraints in a n-window system can be written as follows.

$$\sum_{i \in I} \sum_{j \in J} \delta_{ij}^{k} d_j \beta_i X_{ij} \geq \alpha^k \sum_{j \in J} d_j \ \forall \ k \in 1, 2, \ldots, n \tag{6.1}$$

where δ_{ij}^{k} is the properly defined time-window parameter for time-window k, i.e., $\delta_{ij}^{k} = 1$ if facility i can serve customer j within time-window k, 0 otherwise, and α^k is the required service target for time-window k. Depending upon the problem data in an instance, potentially one among n constraints of type 6.1 will be binding in an optimal solution. The main difficulty for using the single-time-window development

studied extensively in this dissertation for this problem is then: how do we know which one will be binding for each product prior to optimization? Otherwise, as there are multiple sets of δ_{ij} values, one set for each window, dependency relaxation is of no straightforward use for lower bounding the problem with multiple windows.

6.2.3 Multiple Parts and Products

The two multi-part/multi-product models (MPSP-LB and MPMP-LB) can be solved directly using any commercial MIP solver, but as discussed in the previous section, it may turn out to be a computational challenge. Due to the presence of multiple parts, dependency relaxation and the α-boosting-based upper bounding are not very effective here. Specialized techniques must be developed to solve these models more efficiently. Also, if a special technique is available to solve the MPSP-LB (there is no part commonality consideration in this case) then we can build a solution technique for MPMP-LB using ideas similar to those discussed in section 4.3.1.

6.2.4 Customer Centric Service Levels

We focus on the system-wide service level constraints in this dissertation. There are, however, SPL systems in which an individual service level is required for each customer, i.e., each customer has its own service level requirement. The integrated single-part, single-product model with such customer-centric service levels can be handled rather easily by using a part commonality analogy. We can model the problem with customer-centric service levels by introducing a "product" for each customer that shares a common inventory for the part stocked at open facilities.

Define $\lambda_{ij} = \tau d_j X_{ij}$ and $0 \leq Q_{ij} \leq 1$, Consider the following constraints:

$$\begin{aligned} \theta_{ij} &\leq m_{kl}\lambda_{ij} + b_{kl}Q_{ij} + (s_{\max} - l)\sum_{n=1}^{r}(1 - e_{ln})U_{in} \\ &\quad \forall\ i \in I,\ j \in J,\ l \in L, \text{ and } k \in K \end{aligned} \tag{6.2}$$

$$\sum_{j \in J} Q_{ij} = 1\ \forall\ i \in I \tag{6.3}$$

$$\sum_{i \in I} \delta_{ij}\theta_{ij} \geq \alpha_j \tau d_j\ \forall\ j \in J \tag{6.4}$$

These constraints along with the rest of the model formulation will provide us a model to cater customer centric service levels. Constraints (6.2) enable us to perform outer-approximation directly for θ_{ij} variables, using them we can write customer centric service level constraint of the form (6.4). The same idea can be extended to multiple parts and products but it will lot more complex, however, further research may reveal better ideas.

6.2.5 Inventory Sharing

Inventory sharing across same-echelon facilities and lateral transshipments between stocking facilities are becoming more common in SPL systems. This development will reduce stock levels at stocking facilities further and conceivably have an effect on the network decisions. As one modeling alternative here, for example, we can form, prior to optimization, clusters of facilities which participate in *sharing* inventories. We then introduce the cluster fill rate variables, instead of the facility fill rates, that are shared across the facilities in the cluster. Suppose T is the set (indexed by t) of such mutually exclusive clusters and they are decided as parameters of the problem before optimization. Define σ_{it} for all $i \in I$ and $t \in T$ as a binary parameter: $\sigma_{it} = 1$ if facility i belongs to cluster t, 0 otherwise. Using σ_{it}, λ_i, and S_i as defined earlier,

we can write the cluster fill rate as a function of total mean lead time demand (λ_t) and and the total stock level (S_t) in each cluster:

$$\beta_t \quad = \quad \beta(\lambda_t, S_t) \tag{6.5}$$

where $\lambda_t = \sum_{i \in I} \sigma_{it} \lambda_i$, $S_t = \sum_{i \in I} \sigma_{it} S_i$ and β is the proper fill rate function that takes the mean lead time demand as the first parameter, and the stock level as the second parameter. Using a similar Θ-based approximation introduced earlier in the dissertation, we can handle the cluster level allocation of Θ:

$$\Theta_t = \beta_t \lambda_t \tag{6.6}$$

The value of Θ_t can again be determined using outer-approximation and can further be decomposed into θ_{it} values across facilities within cluster t. Furthermore, θ_{it} can be decomposed across multiple customers (θ_{ijt} to write customer centric service level constraints) or multiple products (θ_{ipt} to write product level service level constraints with part commonality). This can be handled again using the same ideas utilized to handle part commonality.

6.2.6 Multi-Echelon Systems

Actual SPL systems typically involve multiple echelons, in which customers are served by stocking facilities at the lowest echelon, which are replenished by upper echelon facilities. We study the problem of the single-echelon network design and inventory stocking in which we locate and stock facilities at the lowest level that will directly serve customers. As a natural and more realistic extension of this effort, one can investigate the multi-echelon systems, where decisions on selecting and stocking

facilities at two or more echelons, allocating customers and lower echelon facilities to upper echelon facilities, are made simultaneously in an integrated model.

6.2.7 Robustness and Sensitivity Analysis

Finally, sensitivity of the cost and solution (location/allocation/stock levels) with respect to the realization of stochastic demands could give us an insight into the robustness of the network and stocking decisions obtained. Research in this regard may involve an analysis with a simulation model which uses the optimization-based solution (network and stock levels). One obvious benefit of such an analysis is we can compare how actual fill rates computed using simulation data match with the fill rate values suggested by MIP solutions. A more insightful analysis would include simulating more realistic scenarios and conditions that are not captured in the mathematical models. This way, we can see how the suggested MIP-based solutions withstand against conditions not explicitly considered in the mathematical models. Such conditions and scenarios to be tested with simulation may include limited stock availabilities at upper echelon facilities and the central warehouse replenishing lowest-echelon facilities (we assume infinite stock levels at the central warehouse), explicit inventory sharing across facilities even across clusters, part substitutions and upgrading, and demand correlation and interdependence.

Appendix

Proof of Proposition 1

$$g(\lambda, s) = \frac{\sum_{n=0}^{s-1} \lambda^{n+1}/n!}{\sum_{n=0}^{s} \lambda^{n}/n!} = \lambda \left(\frac{\sum_{n=0}^{s-1} \lambda^{n}/n!}{\sum_{n=0}^{s} \lambda^{n}/n!} \right)$$

1. We first prove that g is increasing function of λ for given s. For $s = 1$, $g(\lambda, 1) = \frac{\lambda}{1+\lambda}$, which is an increasing function of λ. For $s \geq 2$. Given $\lambda_2 > \lambda_1$, we seek to show that

$$\begin{aligned}
&\lambda_2 \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s} \lambda_2^n/n!} > \lambda_1 \frac{\sum_{n=0}^{s-1} \lambda_1^n/n!}{\sum_{n=0}^{s} \lambda_1^n/n!} \\
\equiv &\frac{\sum_{n=0}^{s} \lambda_2^n/n!}{\sum_{n=0}^{s} \lambda_1^n/n!} < \frac{\lambda_2}{\lambda_1} \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s-1} \lambda_1^n/n!} \\
\equiv &\frac{1 + \sum_{n=1}^{s} \lambda_2^n/n!}{1 + \sum_{n=1}^{s} \lambda_1^n/n!} < \frac{\lambda_2}{\lambda_1} \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s-1} \lambda_1^n/n!}
\end{aligned}$$

   It will be sufficient to show:

$$\begin{aligned}
&\frac{\sum_{n=1}^{s} \lambda_2^n/n!}{\sum_{n=1}^{s} \lambda_1^n/n!} < \frac{\lambda_2}{\lambda_1} \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s-1} \lambda_1^n/n!} \\
\equiv &\frac{\lambda_2 \sum_{n=0}^{s-1} \lambda_2^n/(n+1)!}{\lambda_1 \sum_{n=0}^{s-1} \lambda_1^n/(n+1)!} < \frac{\lambda_2}{\lambda_1} \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s-1} \lambda_1^n/n!}
\end{aligned}$$

$$
\begin{aligned}
&\equiv \frac{\sum_{n=0}^{s-1} \lambda_2^n/(n+1)!}{\sum_{n=0}^{s-1} \lambda_1^n/(n+1)!} < \frac{\sum_{n=0}^{s-1} \lambda_2^n/n!}{\sum_{n=0}^{s-1} \lambda_1^n/n!} \\
&\equiv \left(\sum_{n=0}^{s-1} \frac{\lambda_2^n}{(n+1)!}\right)\left(\sum_{n=0}^{s-1} \frac{\lambda_1^n}{n!}\right) < \left(\sum_{n=0}^{s-1} \frac{\lambda_1^n}{(n+1)!}\right)\left(\sum_{n=0}^{s-1} \frac{\lambda_2^n}{n!}\right) \\
&\equiv \sum_{m=0}^{s-1}\sum_{n=0}^{s-1} \frac{\lambda_1^n \lambda_2^m}{(m+1)!n!} < \sum_{m=0}^{s-1}\sum_{n=0}^{s-1} \frac{\lambda_1^n \lambda_2^m}{(n+1)!m!} \\
&\equiv \sum_{m=0}^{s-1}\sum_{n=0}^{s-1} \frac{\lambda_1^n \lambda_2^m}{m!n!}\left(\frac{1}{m+1} - \frac{1}{n+1}\right) < 0 \\
&\equiv \underbrace{\sum_{s-1>m>n>0} \frac{\lambda_1^n \lambda_2^m}{m!n!}\left(\frac{1}{m+1} - \frac{1}{n+1}\right)}_{T1} \\
&+ \underbrace{\sum_{s-1>n>m>0} \frac{\lambda_1^n \lambda_2^m}{m!n!}\left(\frac{1}{m+1} - \frac{1}{n+1}\right)}_{T2} < 0
\end{aligned}
$$

Defined summations $T1$ and $T2$ both have equal number of terms. Each term in $T1$ is negative since $m > n$, and each term in $T2$ is positive since $m < n$. Hence, $T1 < 0$ and $T2 > 0$. Also, the absolute value of each term in $T1$ (with certain m, n values) is greater than the corresponding term (with the reversed m, n values) in $T2$ because $\lambda_2 > \lambda_1$. This implies that $T1 + T2 < 0$. QED

2. We now show that g is a concave function of λ for given integer $s > 0$. For $s = 1$, $g(\lambda, 1) = \frac{\lambda}{1+\lambda}$, which is a concave function in λ. To show concavity for $s \geq 2$, let

$$
f = \sum_{n=0}^{s} \frac{\lambda^n}{n!}, \quad f' = \sum_{n=1}^{s} \frac{\lambda^{n-1}}{(n-1)!} = \sum_{n=0}^{s-1} \frac{\lambda^n}{n!}
$$
$$
\Rightarrow \quad g(\lambda, s) = \lambda\left(\frac{f'}{f}\right). \tag{6.7}
$$

(For ease of notation, we denote $g(\lambda, s)$ by g_s and $g(\lambda, s-k)$ by g_{s-k}. Also, define $a_s = \lambda^s/s!$ and $a_{s-k} = \lambda^{s-k}/(s-k)!$. Then, $f' = f - a_s$, $f'' =$

$f - a_s - a_{s-1}$)

$$
\begin{aligned}
(6.7) \Rightarrow \quad g_s' &= \frac{ff' + \lambda f f'' - \lambda f'^2}{f^2} \\
&= \frac{\lambda f'/f}{\lambda}\left(1 + \lambda\frac{f''}{f'} - \lambda\frac{f'}{f}\right) \\
&= \frac{g_s}{\lambda}\left(1 + g_{s-1} - g_s\right). \qquad (6.8)
\end{aligned}
$$

$$
\begin{aligned}
\text{And} \quad g_s'' &= \frac{\lambda(g_s' + g_s' g_{s-1} + g_s g_{s-1}' - 2g_s g_s') - \lambda g_s'}{\lambda^2} \\
&= -\frac{1}{\lambda}\frac{d}{d\lambda}(g_s(g_s - g_{s-1})) \\
\Rightarrow \quad g_s'' &= -\frac{1}{\lambda}\left[(g_s - g_{s-1})\frac{dg_s}{d\lambda} + g_s\frac{d}{d\lambda}(g_s - g_{s-1})\right].
\end{aligned}
$$

Since $g_s' > 0$, to show that $g_s'' < 0$ it is sufficient to show:

$$g_s - g_{s-1} > 0, \qquad (6.9)$$

$$g_s' - g_{s-1}' > 0. \qquad (6.10)$$

The condition (6.9) is equivalent to $\lambda\frac{f'}{f} - \lambda\frac{f''}{f'} > 0$

$$
\begin{aligned}
&\equiv \quad \frac{f'}{f} - \frac{f''}{f'} > 0 \\
&\equiv \quad \frac{f - a_s}{f} - \frac{f - a_s - a_{s-1}}{f - a_s} > 0 \\
&\equiv \quad f^2 + a_s^2 - 2fa_s - f^2 + fa_s + fa_{s-1} > 0 \\
&\equiv \quad a_s^2 - fa_s + fa_{s-1} > 0 \\
&\equiv \quad a_s - f\left(1 - \frac{a_{s-1}}{a_s}\right) > 0 \\
&\equiv \quad a_s - f\left(1 - \frac{s}{\lambda}\right) > 0
\end{aligned}
$$

$$
\begin{aligned}
\equiv \quad & \frac{\lambda^s}{s!} - \sum_{n=0}^{s} \frac{\lambda^n}{n!} + s \sum_{n=0}^{s} \frac{\lambda^{n-1}}{n!} > 0 \\
\equiv \quad & -\sum_{n=0}^{s-1} \frac{\lambda^n}{n!} + \frac{s}{\lambda} + s \sum_{n=1}^{s} \frac{\lambda^{n-1}}{n!} > 0 \\
\equiv \quad & -\sum_{n=0}^{s-1} \frac{\lambda^n}{n!} + \frac{s}{\lambda} + s \sum_{n=0}^{s-1} \frac{\lambda^n}{(n+1)!} > 0 \\
\equiv \quad & \frac{s}{\lambda} + \sum_{n=0}^{s-1} \frac{\lambda^n}{n!} \left(\frac{s}{n+1} - 1 \right) > 0 \\
\equiv \quad & \frac{s}{\lambda} + \sum_{n=0}^{s-2} \frac{\lambda^n}{n!} \left(\frac{s}{n+1} - 1 \right) > 0
\end{aligned}
$$

The last inequality holds because $\left(\frac{s}{n+1} - 1\right) > 0$ for $n \leq s-2$. Using (6.8), we rewrite (6.10) as follows:

$$
\begin{aligned}
& \frac{g_s}{\lambda} (1 + g_{s-1} - g_s) - \frac{g_{s-1}}{\lambda} (1 + g_{s-2} - g_{s-1}) > 0 \\
\equiv \quad & g_s (1 + g_{s-1} - g_s) - g_{s-1} (1 + g_{s-2} - g_{s-1}) > 0 \qquad \text{since } \lambda > 0 \\
\equiv \quad & 2g_{s-1} - g_s - g_{s-2} > 0 \qquad \text{since } g_s > g_{s-1} \\
\equiv \quad & 2\frac{a_{s-1}}{f'} - \frac{a_s}{f} - \frac{a_{s-2}}{f''} < 0 \qquad (6.11)
\end{aligned}
$$

We now observe that a_s/f is the Erlang loss function: $B(\lambda, s) = \frac{a_s}{f} = \left(\frac{\lambda^s}{s!}\right) \Big/ \sum_{n=0}^{s} (\lambda^n/n!)$. Now (6.11) is equivalent to: $2B(\lambda, s-1) > B(\lambda, s) + B(\lambda, s-2)$. Since the Erlang loss formula is known to be a convex function of $s > 0$ for any $\lambda > 0$ (Messerli 1972, Jagers and van Doorn 1986), this condition is satisfied, which proves that (6.10) holds. QED

Proof of Proposition 3

The left hand side of constraint (3.11) is non-decreasing in $\Theta_i \ \forall \ i \in I$. Given a solution of RBM, if there is slack in constraints (3.18), we can transfer this slack to constraints (3.11) and hence increase the achieved service level without increasing the cost. If the optimal solution of RBM satisfies constraints (3.18) with equality then it is also an optimal to SBM. Else we can construct another optimal solution of RBM by inflating β_i's such that constraints (3.18) are satisfied with equality in the constructed solution. The constructed solution will have the same cost and remains feasible to RBM, moreover, it will also remain feasible and optimal to SBM. This construction when required can be done in linear time. QED

Proof of Proposition 4 ($\lim_{\lambda\to\infty} g(\lambda, s) = s$ for given s)

$$\begin{aligned} g(\lambda, s) &= \frac{\lambda + \lambda^2/1! + \lambda^3/2! + \ldots + \lambda^s/(s-1)!}{1 + \lambda + \lambda^2/2! + \ldots + \lambda^s/s!} \\ &= \frac{\lambda^{-(s-1)} + \lambda^{-(s-2)}/1! + \lambda^{-(s-3)}/2! + \ldots + 1/(s-1)!}{1/\lambda^{-s} + \lambda^{-(s-1)} + \lambda^{-(s-2)}/2! + \ldots + 1/s!} \\ \Rightarrow \quad \lim_{\lambda\to\infty} g(\lambda, s) &= \frac{1/(s-1)!}{1/s!} = s!/(s-1)! = s. \text{ QED} \end{aligned}$$

Proof of Proposition 5

β is a decreasing function of λ for any given s. Since $\lambda_i^{\text{in}} \leq \lambda_i$, we have $\beta_i(\lambda_i^{\text{in}}) \geq \beta_i(\lambda_i)$ for all i. This implies:

$$\sum_{i\in I}\sum_{j\in J} \delta_{ij} d_j \beta_i(\lambda_i^{\text{in}}) X_{ij} \geq \sum_{i\in I}\sum_{j\in J} \delta_{ij} d_j \beta_i(\lambda_i) X_{ij} \geq \alpha \sum_{j\in J} d_j,$$

which shows that the new model is a relaxation. QED

Obtaining Valid Inequalities in Binary Representation Variables

Here we show that $Y_i \leq \sum_{n=1}^{r} U_{in},\ \forall\ i \in I$ are valid and are lifted versions of original constraints $S_i \geq X_{ij},\ \ \forall\ i \in I$. Stock levels are represented using U_{in}'s as $S_i = \sum_{n=1}^{r} 2^{n-1} U_{in}\ \forall\ i \in I$. The original constraints $S_i \geq X_{ij}$ are equivalent to: $\sum_{n=1}^{r} 2^{n-1} U_{in} \geq X_{ij},\ \forall\ i \in I$ and $j \in J$ which along with $X_{ij} \leq Y_i$, for all i and j imply:

$$\sum_{n=1}^{r} 2^{n-1} U_{in} \geq Y_i,\ \forall\ i \in I \tag{6.12}$$

since, $Y_i = \max_{j \in J}\{X_{ij}\}$. Further, inequalities (6.12) can be lifted since the coefficients of U_{in}'s do not matter as U_{in}'s themselves represent whether or not a facility is being stocked. Finally, we obtain:

$$\sum_{n=1}^{r} U_{in} \geq Y_i,\ \forall\ i \in I. \tag{6.13}$$

Bibliography

Anupindi, R., S. Tayur. 1998. Managing stochastic multi-product systems: model, measures and analysis. *Operations Research* **46**(3s) 98–111.

Baker, K. R., M. J. Magazine, H. L. W. Nuttle. 1986. The effect of commonality on safety stock in a simple inventory model. *Man. Sci.* **32**(8) 982–988.

Baker, K.R. 1985. Safety stocks and component commonality. *Jour. Oper. Man.* **6** 13–23.

Barahona, F., D. Jensen. 1998. Plant location with minimum inventory. *Math. Prog.* **83** 101–111.

Candas, M.F., E. Kutanoglu. 2006a. Benefits of considering inventory in service parts logistics network design problems with time-based service constraints. *Forthcoming in IIE Transactions* .

Candas, M.F., E. Kutanoglu. 2006b. Column generation for integrated inventory and network design problem with system-wide service levels. *Working Paper* .

Cheung, K.L., W.H. Hausman. 1995. Multiple failures in a multi-item spares inventory model. *IIE Transactions* **27** 171–180.

Cohen, M.A., C. Cull, H.L. Lee, D. Willen. 2000a. Saturn's supply chain innovation: High value in after sales service. *MIT Sloan Man. Rev.* **41**(4, Summer) 93–101.

Cohen, M.A., P. Kamesam, P.R. Kleindorfer, H.L. Lee, A. Tekerian. 1990. OPTIMIZER: IBM's multi-echelon inventory system for managing service logistics. *Interfaces* **20**(1) 65–82.

Cohen, M.A., P.R. Kleindorfer, H.L. Lee. 1988. Service constrained (s,S) inventory systems with priority demand classes and lost sales. *Man. Sci.* **34**(4) 482–499.

Cohen, M.A., P.R. Kleindorfer, H.L. Lee. 1989. Near-optimal service constrained stocking policies for spare parts. *Oper. Res.* **37**(1) 104–117.

Cohen, M.A., P.R. Kleindorfer, H.L. Lee, D.F. Pyke. 1992. Multi-item service constrained (s,S) policies for spare parts logistics systems. *Naval Research Logistics* **39** 561–577.

Cohen, M.A., H.L. Lee. 1988. Strategic analysis of integrated production-distribution systems: Models and methods. *Oper. Res.* **36**(2) 216–228.

Cohen, M.A., H.L. Lee. 1990. Out of touch with customer needs: Spare parts and after sales service. *MIT Sloan Man. Rev.* **Winter** 55–66.

Cohen, M.A., H.L. Lee, C. Cull, D. Willen. 2000b. Making the supply chain work: How to deliver value in after sales service logistics. *MIT Sloan Man. Rev.* **41**(4) 93–101.

Cohen, M.A., Y-S. Zheng, V. Agrawal. 1997. Service parts logistics: A benchmark analysis. *IIE Transactions* **29** 627–639.

Cohen, M.A., Y-S. Zheng, Y. Wang. 1999. Identifying opportunities for improving teradyne's service parts logistics system. *Interfaces* **29**(4) 1–18.

Collier, D.A. 1981. The measurement and operations benefits pf component part commonality. *Decision Science* **12**(1) 85–96.

Collier, D.A. 1982. Aggregate safety stock levels and component part commonality. *Man. Sci.* **88**(11) 1296–1303.

Daskin, M.S. 1995. *Network and Discrete Location: Models, Algorithms and Methods*. John Wiley & Sons, Inc., New York.

Daskin, M.S., C.R. Coullard, Z.-J.M. Shen. 2002. An inventory-location model: Formulation, solution algorithm and computational results. *Annals of Operations Research* **110** 13–106.

Daskin, M.S., L.V. Snyder, R.T. Berger. 2005. Facility location in supply chain design. A. Langevin, D. Riopel, eds., *Forthcoming in Logistics Systems: Design and Operations*. Kluwer.

Drezner, Z. 1995. *Facility Location: A survey of applications and methods*. Springer, New York, NY.

Duran, M. A., I. E. Grossmann. 1986. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Prog.* **36** 307–339.

Geoffrion, A.M. 1979. Making better use of optimization capability in distribution system planning. *AIIE Transactions* **11**(2) 96–108.

Geoffrion, A.M. 1989. Integrated modeling systems. *Computer Sci. Eco. and Man.* **2**(1) 3–15.

Geoffrion, A.M., R.F. Powers. 1980. Facility location analysis is just the beginning. *Interfaces* **10**(2) 22–30.

Geoffrion, A.M., T.J. Van Roy. 1979. Caution: Common sense planning methods can be hazardous to your corporate health. *Sloan Man. Rev.* **20**(4) 31–42.

Gerchak, Y., M.J. Magazine, A.B. Gamble. 1988. Component commonality with service level requirements. *Man. Sci.* **34** 753–760.

Goldberg, J.R., L. Paz. 1991. Locating emergency vehicle bases when service time depends on call location. *Trans. Sci.* **25** 264–280.

Graves, S.C., A.H.G. Rinnooy Kan, P.H. Zipkin, eds. 1993. *Handbooks of Operations Research*, *Logistics for Production and Inventory*, vol. 4. North-Holland, Amsterdam.

Grossmann, I.E., J. Viswanathan, A. Vecchietti, R. Raman, E. Kalvelagen. 2001. *DICOPT: A Discrete Continuous Optimization Package*. GAMS Developement Corp. Available at: http://www.gams.com/dd/docs/solvers/dicopt.pdf.

Guignard, M., S. Kim. 1987. Lagrangean decomposition: A model yielding strong lagrangean bounds. *Mathematical Programming* **39** 215–228.

Hausman, W. H., H. L. Lee, A. X. Zhang. 1998. Joint demand fulfillment probability in a multiitem inventory system with independent order-up-to policies. *European Journal of Operational Research* **109** 646–659.

Hopp, W.J., M.L. Spearman. 2000. *Factory Physics, 2 edition*. McGraw-Hill - Irwin, Burr Ridge, IL.

Iyoob, I., E. Kutanoglu. 2006. Integrated logistics network design and inventory stocking with customer-dedicated facilities. *Working Paper* .

Jagers, A., E. van Doorn. 1986. On the continued Erlang loss function. *Operations Research Letters* **5** 43–46.

Johnson, L.A., D.C. Montgomery. 1974. *Operations Research in Production Planning, Scheduling, and Inventory Control*. John Wiley & Sons., New York, NY.

Kalvalagen, E. 2002. Interpolation with gams. *GAMS help: http://www.gams.com/ erwin* .

Magnanti, T.L., R.T. Wong. 1984. Network design and transportation planning: Model and algorithms. *Trans. Sci.* **18** 1–55.

Messerli, E. J. 1972. Proof of a convexity property of the Erlang B formula. *Bell System Technical Journal* **51**(4) 951–953.

Miranda, P.A., R.A. Garrido. 2004. Incorporating inventory control decisions into a strategic distribution network design model with stochastic demand. *Trans. Res. Part E* **40** 183–207.

Mirchandani, P., A.K. Mishra. 2002. Component commonality: models with product-specific service constraints. *Prod. Oper. Man.* **11**(2(Summer)) 199–215.

Muckstadt, J.A. 1973. A model for multi-item, multi-echelon, multi-indenture inventory system. *Man. Sci.* **20** 472–481.

Muckstadt, J.A. 2005. *Analysis and Algorithms for Service Parts Supply Chains*. Springer, New York, NY.

Muckstadt, J.A., L.J. Thomas. 1980. Are multi-echelon inventory methods worth implementing in systems with low-demand-rate items? *Man. Sci.* **26** 483–494.

Nahmias, S. 1981. Managing repairable inventory systems: a review. *TIMS Studies in Management Science* **16** 253–277.

Nozick, L.K. 2001. The fixed charge facility location problem with coverage restrictions. *Trans. Res. Part E* **37** 281–296.

Nozick, L.K., M.A. Turnquist. 1998. Integrating inventory impacts into a fixed-charge model for locating distribution centers. *Trans. Res. Part E* **34**(3) 173–186.

Nozick, L.K., M.A. Turnquist. 2001. Inventory, transportation, service quality and the location of distribution centers. *Eur. Jour. Oper. Res.* **129** 362–371.

Ouyang, L.Y., K.S. Wu. 1997. Mixture inventory model involving variable lead time with a service level constraint. *Comp. Oper. Res.* **24**(9) 875–882(8).

Rustenburg, W.D., G.J. van Houtum, W.H.M. Zijm. 2001. Spare parts management at complex technology-based organizations: An agenda for research. *International Jour. Prod. Eco.* **71** 177–193.

Schwarz, L.B. 1981. Physical distribution: The analysis of inventory and location. *AIIE Trans.* **13** 138–150.

Shen, Z.-J.M., C. Coullard, M.S. Daskin. 2003. A joint location-inventory model. *Trans. Res.* **37**(1) 40–55.

Sherbrooke, C.C. 1968. METRIC: A multi-echelon technique for recoverable item control. *Oper. Res.* **16** 122–141.

Sherbrooke, C.C. 1986. VARI-METRIC: Improved approximation for multi-indenture, multi-echelon availability models. *Oper. Res.* **34** 311–319.

Sherbrooke, C.C. 1992. *Optimal Inventory Modeling of Systems*. John Wiley and Sons, New York, NY.

Snyder, L.V., M.S. Daskin. 2005. Reliability models for facility location: The expected failure cost case. *Trans. Res.* **39**(3) 400–416.

Song, J.-S. 1998a. On the order fill rate in a multi-item, base-stock inventory system. *Oper. Res.* **46**(6).

Song, Jing-Sheng. 1998b. On the order fill rate in a multi-item, base-stock inventory system. *Operations Research* **46**(6) 831–845.

Srinivasan, R., R. Jayaraman, J. A. Rappold, R. O. Roundy, S. Tayur. 1992. Procurement of common components in a stochastic environment. *TR Cornell Univ.* (1277).

Thonemann, U.W., M.L. Brandeau. 2000. Optimal commonality in component design. *Oper. Res.* **48**(1) 1–19.

Verrijdt, J.H.C.M., A.G. de Kok. 1995. Distribution planning for a divergent n-echelon network without intermediate stocks under service restrictions. *International Jour. Prod. Eco.* **38**(2-3) 225–243.

Verrijdt, J.H.C.M., A.G. de Kok. 1996. Distribution planning for a divergent depotless two-echelon network under service constraints. *Eur. Jour. Oper. Res.* **89** 341–354.

Wagner, H.M. 1974. The design of production and inventory systems for multifacility and multiwarehouse companies. *Oper. Res.* **22**(2) 278–291.

Zipkin, P. 2000. *Fundemantals of Inventory Management*. McGraw Hill - Irwin, Boston, MA.

Vita

Vishv Jeet was born in Bareilly, India, on December 24, 1976, of Malti Devi Agarwal and Om Prakash Agarwal. Starting at the age of 4, he studied at the Laxminarayan Junior High School in Bareilly for the first 7 years and for the next 6 years he studied at Government Inter College, Bareilly, where he completed his education up to 11th standard. He was admitted to Government Polytechnic Nainital in 1992 where he got his diploma in Electrical Engineering specializing in Electric Traction. He then attended Indian Institute of Technology at Bombay from 1996 to 2001 to complete his Bachelors and Masters degrees in Mechanical Engineering and Computer Integrated Manufacturing, respectively. Thereafter, he was a system analyst with Citicorp in Bombay for a year contributing towards the development of a security armour for Citibank's online operations.

In August 2002, he joined the University of Texas at Austin to pursue his doctoral studies. He was awarded the best doctoral dissertation proposal support award by Supply Chain and Logistics Engineering Center at University of Florida, Gainesville in February 2005. He received Doctor of Philosophy in Operations Research in December 2006. Since July 2006, he has been working with a Austin based start-up company called Gravitant, Inc.

Permanent Address: c/o Lalit Kumar Agarwal

54 Mechanier Road,

Bareilly 243003,

Uttar Pradesh, INDIA

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.